

```
UUU      UUU VVV      VVV      111      RRRRRRRRRRRR      0000000000      MMM      MMM
UUU      UUU VVV      VVV      111      RRRRRRRRRRRR      0000000000      MMM      MMM
UUU      UUU VVV      VVV      111      RRRRRRRRRRRR      0000000000      MMM      MMM
UUU      UUU VVV      VVV      111111      RRR      RRR      000      000      MMMMMM      MMMMMM
UUU      UUU VVV      VVV      111111      RRR      RRR      000      000      MMMMMM      MMMMMM
UUU      UUU VVV      VVV      111111      RRR      RRR      000      000      MMMMMM      MMMMMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUUUUUUUUUUUUUUUUU      VVV      1111111111      RRR      RRR      0000000000      MMM      MMM
UUUUUUUUUUUUUUUUUU      VVV      1111111111      RRR      RRR      0000000000      MMM      MMM
UUUUUUUUUUUUUUUUUU      VVV      1111111111      RRR      RRR      0000000000      MMM      MMM
```



```
FFFFFFFFF  IIIII  LL  EEEEEEEEE  RRRRRRR  DDDDDDD  UU  UU  VV  VV  11
FFFFFFFFF  IIIII  LL  EEEEEEEEE  RRRRRRR  DDDDDDD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  1111
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  1111
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FFFFFFFFF  II    LL  EEEEEEE  RRRRRRR  DD    DD  UU  UU  VV  VV  11
FFFFFFFFF  II    LL  EEEEEEE  RRRRRRR  DD    DD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FF        II    LL  EE          RR      RR  DD    DD  UU  UU  VV  VV  11
FF        IIIII  LLLLLLLLL  EEEEEEEEE  RR      RR  DDDDDDD  UUUUUUUUU  VV  VV  111111
FF        IIIII  LLLLLLLLL  EEEEEEEEE  RR      RR  DDDDDDD  UUUUUUUUU  VV  VV  111111
                                     ....
                                     ....
                                     ....
                                     ....

LL  IIIII  SSSSSSS
LL  IIIII  SSSSSSS
LL  II    SS
LL  II    SS
LL  II    SS
LL  II    SS
LL  II    SSSSSS
LL  II    SSSSSS
LL  II    SS
LL  II    SS
LL  II    SS
LL  IIIII  SSSSSSS
LL  IIIII  SSSSSSS
```

(1)	59	DECLARATIONS
(2)	168	FIL\$OPENFILE - RETURN FILE HEADER AND STATISTICS BLOCK
(3)	448	FIL\$CACHE_INIT - INIT FILEREAD CACHE
(4)	540	FIL\$CACHE_TRUNC - TRUNCATE FILEREAD CACHE
(6)	636	STORE3DIGITS - STORE 3 ASCII DIGITS
(7)	671	FORMDIRSTRING - GET A DIRECTORY STRING
(8)	738	MOUNT - MOUNT THE VOLUME, INIT FOR FILE LOOKUP
(9)	846	FINDFILID - FIND FILE ID FOR SPECIFIED FILE
(10)	1106	FIL\$FINDFILID - STRUCTURE LEVEL 2
(11)	1241	READ DIR LBN - READ NEXT DIRECTORY LBN
(12)	1284	RDCHKFILHDR - READ AND CHECK FILE HEADER
(13)	1429	READVBN, WRITEVBN - READ/WRITE VIRTUAL BLOCK
(14)	1524	INIRTRVPTRSCAN - INITIALIZE RETRIEVAL POINTER SCAN
(15)	1552	GETRTRVPTR - CONVERT NEXT RETRIEVAL POINTER
(16)	1637	STATBLK - GET FILE STATISTICS BLOCK
(17)	1751	FIL\$CHKFILHDR - CHECK FILE HEADER VALIDITY
(18)	1814	CHECKSUM - VALIDATE A CHECKSUM



```
00000001 0000 1 BOOT_UV1_SWITCH = 1 ; Build Micro-VAX I bootstrap emulator
00000001 0000 2 PQ == 1
0000 3 .NLIST CND
0000 4 .TITLE FILEREADUV1 - MICRO-VAX I FILES-11 LEVEL 2 FILE READING ROUTINES
0000 5 .IDENT 'V03-003'
0000 6
0000 7
0000 8
0000 9
0000 10 :
0000 11 :*****
0000 12 :*
0000 13 :* COPYRIGHT (c) 1978, 1980, 1982, 1983 BY
0000 14 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 15 :* ALL RIGHTS RESERVED.
0000 16 :*
0000 17 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 18 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 19 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 20 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 21 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 22 :* TRANSFERRED.
0000 23 :*
0000 24 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 25 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 26 :* CORPORATION.
0000 27 :*
0000 28 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 29 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 30 :*
0000 31 :*
0000 32 :*****
0000 33 :
0000 34 :++
0000 35 : FACILITY: USER CALLABLE PROCEDURES
0000 36 :
0000 37 : ABSTRACT:
0000 38 :
0000 39 : THIS SET OF ROUTINES PROVIDES THE CAPABILITY OF 'OPENING' AND
0000 40 : READING FILES BY FILE NAME FROM A FILES11 STRUCTURE LEVEL 2 VOLUME.
0000 41 : THERE IS NO MULTI-VOLUME SUPPORT, AND MULTI-HEADER SUPPORT IS LIMITED
0000 42 : TO RETURNING THE CORRECT FILE SIZE IN THE STATBLK.
0000 43 :
0000 44 : ENVIRONMENT: USER MODE
0000 45 :
0000 46 : AUTHOR: PETER H. LIPMAN , CREATION DATE: 14-DEC-76
0000 47 :
0000 48 : MODIFIED BY:
0000 49 :
0000 50 : V03-003 KDM0070 Kathleen D. Morse 11-Aug-1983
0000 51 : Create Micro-VAX I version of FILEREAD. It has no CMPC5
0000 52 : instructions as those require software emulation.
0000 53 :
0000 54 : V03-002 KDM0041 Kathleen D. Morse 14-Apr-1983
0000 55 : Remove the ODS-1 structure level support.
0000 56 :
0000 57 :--
```



```
0000 59 .SBTTL DECLARATIONS
0000 60 :
0000 61 : INCLUDE FILES:
0000 62 :
0000 63 .nocross
0000 64 $DIRDEF ;DIRECTORY ENTRY OFFSET DEFINITIONS
0000 65 $FATDEF ;RECORD ATTRIBUTE AREA DEFINITIONS
0000 66 $FH2DEF ;FILE HEADER DEFINITIONS, LEVEL 2
0000 67 $FM2DEF ;MAP AREA, LEVEL 2
0000 68 $FIDDEF ;FILE ID OFFSET DEFINITIONS
0000 69 $HM2DEF ;HOME BLOCK DEFINITIONS, LEVEL 2
0000 70 $IODEF ;I/O DEFINITIONS
0000 71 $PSLDEF ;PROCESSOR STATUS LONG WORD DEFINITIONS
0000 72 $SSDEF ;SYSTEM SERVICE DEFINITIONS
0000 73 :
0000 74 : MACROS:
0000 75 :
0000 76 .MACRO READVBN CHAN,VBN,BUFADR,HDRADR
0000 77 .LIST MEB
0000 78 PUSHAL HDRADR
0000 79 PUSHAL BUFADR
0000 80 PUSHL VBN
0000 81 PUSHL CHAN
0000 82 CALLS #4,W^FIL$READVBN
0000 83 .NLIST MEB
0000 84 .ENDM READVBN
0000 85
0000 86 .MACRO READLBN CHAN,VBN,BUFADR
0000 87 .LIST MEB
0000 88 ROTL #9,#1,-(SP)
0000 89 MOVZWL #10$ READLBLK,-(SP)
0000 90 PUSHAL BUFADR
0000 91 PUSHL VBN
0000 92 PUSHL CHAN
0000 93 CALLS #5,W^FIL$RDWRTLBN
0000 94 .NLIST MEB
0000 95 .ENDM READLBN
0000 96 .cross
0000 97 :
0000 98 : EQUATED SYMBOLS:
0000 99 :
0000 100
000001FE 0000 101 FH2$W_VBNOFFSET = FH2$W_CHECKSUM ;SAVE INDEX FILE VBN OFFSET
0000 102 ;IN THIS PLACE IN INDEX FILE HEADER
00000009 0000 103 ASSUME FH2$C_LEVEL2@-8 EQ 2
0000000A 0000 104 FH2$V_LEVEL2 = 9
0000 105 FH2$V_BIGFILNUM = 10 ;IF SET USE HIGH 8 BITS OF FILE ID RVN
0000 106 ;FIELD AS FILE NUMBER EXTENSION
0000 107 ;BIT IS PLACED IN FH2$W_STRUCLEV
0000 108 ;BY THE FIL$MOUNT CODE
00000001 0000 109
0000 110 FIL$C_CACHE_ID = 1 ;VERSION OF THE FILEREAD CACHE
0000 111 :
0000 112 : OFFSETS INTO HEADER PORTION OF THE FILEREAD CACHE
0000 113 :
0000 114 $OFFSET 0,POSITIVE,<-
0000 115 <FIL$W_CACHE_ID,2>,- ;CACHE IDENT, AND WRITE INTERLOCK BIT
```

```

0000 116      <2>,-                               ;SPARE
0000 117      FILSL_DIROFF,-                     ;OFFSET IN BYTES TO DIRECTORY CACHE
0000 118      FILSL_DIRNXT,-                     ;NEXT OFFSET TO ALLOCATE DIR CACHE ENTRY
0000 119      <FILSC_DIRMAX,0>,-                 ;MAX OFFSET FOR DIR CACHE
0000 120      FILSL_LBNOFF,-                     ;OFFSET IN BYTES TO BEGIN OF LBN CACHE
0000 121      FILSL_LBNNXT,-                     ;NEXT OFFSET TO ALLOCATE LBN CACHE
0000 122      FILSL_LBNMAX,-                     ;MAX OFFSET FOR LBN CACHE
0000 123      <FILSA_IXFHDR,512>,-               ;INDEX FILE HEADER
0000 124      <FILSC_SIZE,0>-                     ;START OF DIRECTORY CACHE
0000 125      >
0000      FILSW_CACHE_ID:
0004      FILSL_DIROFF:
0008      FILSL_DIRNXT:
000C      FILSL_DIRMAX:
000C      FILSL_LBNOFF:
0010      FILSL_LBNNXT:
0014      FILSL_LBNMAX:
0018      FILSA_IXFHDR:
0218      FILSC_SIZE:
0000 126      ;
0000 127      ; OFFSETS INTO DIRECTORY CACHE ENTRIES
0000 128      ;
0000 129      $OFFSET 0,POSITIVE,<-
0000 130      <FILSA_DIR_FID,6>,-                 ;DIRECTORY ID
0000 131      <FILST_DIR_NAM,10>,-                ;COUNTED NAME OF DIRECTORY - NO ".DIR"
0000 132      <FILSQ_DIR_HDR,0>,-                 ;DIRECTORY HEADER INFORMATION
0000 133      <FILSW_DIR_BKCNT,2>,-               ;SIZE IN BLOCKS OF DIRECTORY FILE
0000 134      <FILSB_DIR_LVL,1>,-                ;STRUCTURE LEVEL OF DIRECTORY
0000 135      <,1>,-                             ;SPARE BYTE
0000 136      FILSL_DIR_LBN,-                     ;STARTING LBN OF DIRECTORY
0000 137      FILSL_DIR_BFOFF,-                  ;OFFSET TO DIR LBN BUFFER
0000 138      <FILSQ_DIR_BFCNT,2>,-              ;SIZE IN BLOCKS OF DIR LBN BUFFER
0000 139      <FILSA_DIR_OFID,6>,-               ;OUTPUT FILE ID FROM LOOKUP
0000 140      <FILSC_DIR_SIZE,0>-                 ;SIZE OF DIRECTORY CACHE ENTRY
0000 141      >
0000      FILSA_DIR_FID:
0006      FILST_DIR_NAM:
0010      FILSQ_DIR_HDR:
0010      FILSW_DIR_BKCNT:
0012      FILSB_DIR_LVL:
0014      FILSL_DIR_LBN:
0018      FILSL_DIR_BFOFF:
001C      FILSW_DIR_BFCNT:
001E      FILSA_DIR_OFID:
0024      FILSC_DIR_SIZE:
0000 142      ;
0000 143      ; MAKE THESE GLOBAL SO THAT A CACHE SIZE CAN BE PROPERLY CALCULATED
0000 144      ; THE CALCULATION IS:
0000 145      ;
0000 146      ;     FILSC_SIZE + (DIRCNT * FIILSC_DIR_SIZE) + <LBNCNT * 512>
0000 147      ;
0000 148      ;     .GLOBAL FILSC_SIZE,FILSC_DIR_SIZE
0000 149      ;
0000 150      ; DEFINE THE FOLLOWING WEAK REFERENCES, THEY NEED NOT BE PRESENT
0000 151      ;
0000 152      .WEAK    FILSQ_CACHE                     ;DESCRIPTOR FOR FILEREAD CACHE
0000 153      .WEAK    FILSGT_DDDEV                  ;ASCIC DEFAULT DEVICE NAME STRING

```



FILEREADUV1  
V03-003

M 7  
- MICRO-VAX 1 FILES-11 LEVEL 2 FILE READ 10-AUG-1984 18:05:11 VAX/VMS Macro V04-00  
DECLARATIONS 9-JUL-1984 11:44:50 FILEREAD.MAR;1

Page 4  
(1)

```
0000 154 .WEAK FIL$GT_TOPSYS ;ASCIC TOP LEVEL SYSTEM DIRECTORY
0000 155 :
0000 156 : OWN STORAGE:
0000 157 :
0000 158 :
00000000 159 .PSECT YFILEREAD,BYTE,EXE
0000 160
0000 161 FIL_GQ_CACHE:
00000000' 0000 162 .ADDRESS FIL$GQ_CACHE
0000 163 FIL_GT_DDDEV:
00000000' 0004 164 .ADDRESS FIL$GT_DDDEV
0000 165 FIL_GT_TOPSYS:
00000000' 0008 166 .ADDRESS FIL$GT_TOPSYS
```

```

000C 168 .SBTTL FIL$OPENFILE - RETURN FILE HEADER AND STATISTICS BLOCK
000C 169 :++
000C 170 : FUNCTIONAL DESCRIPTION:
000C 171 :
000C 172 : THE OPENFILE ROUTINE ACCEPTS A FULL FILE NAME IN THE FORMAT
000C 173 : DEV:[DIR]FILE.TYP;VERSION.
000C 174 : IT ASSIGNS AND RETURNS A CHANNEL, READS THE FILE HEADER, RETURNS THE
000C 175 : STATISTICS BLOCK, AND OPTIONALLY RETURNS THE RETRIEVAL POINTERS IN
000C 176 : A NORMALIZED (LONG WORD COUNT, LONG WORD LBN) FORMAT.
000C 177 : THE DIRECTORY MAY BE IN ANY OF THE STANDARD FORMATS:
000C 178 : [10,40], [010040], [ABCDEFGH1], OR WITH < AND > REPLACING [ AND ].
000C 179 : VERSION MAY BE ZERO IN WHICH CASE THE HIGHEST VERSION IS FOUND
000C 180 :
000C 181 : CALLING SEQUENCE:
000C 182 :
000C 183 : CALLG  ARGLIST,FIL$OPENFILE
000C 184 :
000C 185 : INPUT PARAMETERS:
000C 186 :
000C 187 :     CHANADR(AP)      =      ADDRESS TO RETURN CHANNEL
000C 188 :     FILNAM(AP)      =      ADDRESS OF 2 LONG WORD FILE NAME STRING DESCRIPTOR
000C 189 :                             1 - SIZE OF STRING
000C 190 :                             2 - ADDRESS OF STRING
000C 191 :                             DB1:[10,40]FILST.EXE
000C 192 :     IXFHDR(AP)      =      ADDRESS OF 512 BYTE BUFFER TO BE USED FOR
000C 193 :                             THE INDEX FILE HEADER
000C 194 :     FILHDR(AP)      =      ADDRESS OF 512 BYTE BUFFER TO RETURN FILE HEADER
000C 195 :     STATBLK(AP)     =      ADDRESS OF 2 LONG WORD BLOCK IN WHICH THE
000C 196 :                             FOLLOWING WILL BE RETURNED
000C 197 :                             1 - LOGICAL BLOCK NUMBER OF FIRST BLOCK OF
000C 198 :                             FILE OR 0 IF FILE IS NOT CONTIGUOUS
000C 199 :                             2 - SIZE OF FILE IN BLOCKS
000C 200 :     RTRVPTRLEN(AP)  =      ADDRESS TO RETURN THE NUMBER OF
000C 201 :                             BYTES OF RETRIEVAL POINTERS STORED
000C 202 :                             ***** OPTIONAL PARAMETER *****
000C 203 :     RTRVPTRBUF(AP)  =      ADDRESS OF RETRIEVAL POINTER
000C 204 :                             BUFFER DESCRIPTOR. THIS PARAMETER
000C 205 :                             IS PRESENT IF AND ONLY IF
000C 206 :                             RTRVPTRLEN IS PRESENT.
000C 207 :                             THE RETRIEVAL POINTERS ARE RETURNED IN
000C 208 :                             THE FORM 32 BIT BLOCK COUNT, 32 BIT LBN
000C 209 :                             A ZERO BUFFER DESCRIPTOR ADDRESS OR A
000C 210 :                             ZERO BUFFER ADDRESS MEANS DON'T
000C 211 :                             RETURN RETRIEVAL POINTER INFO
000C 212 :
000C 213 : IMPLICIT INPUTS:
000C 214 :
000C 215 :     NONE
000C 216 :
000C 217 : OUTPUT PARAMETERS:
000C 218 :
000C 219 :     RO = SYSTEM STATUS CODE
000C 220 :
000C 221 : IMPLICIT OUTPUTS:
000C 222 :
000C 223 :     NONE
000C 224 :

```



```

000C 225 : COMPLETION CODES:
000C 226 :
000C 227 :     $$$_NORMAL          SUCCESSFUL COMPLETION
000C 228 :     $$$_NOSUCHFILE     FAILED TO FIND DIRECTORY OR FILE
000C 229 :     $$$_BADFILENAME      SYNTAX ERROR IN DIRECTORY OR FILE NAME STRING
000C 230 :
000C 231 : THE FOLLOWING COMPLETION CODES INDICATE FILE STRUCTURE PROBLEMS
000C 232 :
000C 233 :     $$$_BADCHKSUM          CHECKSUM ERROR IN HOME BLOCK, INDEX FILE HEADER
000C 234 :                             DIRECTORY FILE HEADER OR FILE HEADER
000C 235 :     $$$_BADFILEHDR         FILE HEADER CONSISTENCY CHECK FAILED FOR
000C 236 :                             INDEX FILE, DIRECTORY FILE, OR DESIRED FILE
000C 237 :     $$$_FILESTRUCT         HOME BLOCK INDICATES THAT THIS VOLUME
000C 238 :                             CONTAINS A NON-SUPPORTED FILE STRUCTURE
000C 239 :                             OR POSSIBLY THE HOMEBLOCK IS GARBAGE
000C 240 :
000C 241 : SIDE EFFECTS:
000C 242 :
000C 243 :     NONE
000C 244 :
000C 245 : EQUATED SYMBOLS
000C 246 :
000C 247 :     OFFSETS FROM AP
000C 248 :
000C 249 :     ARGCNT                 = 0
000C 250 :     CHANADR                = 4
000C 251 :     FILNAM                 = 8
000C 252 :     IXFHDR                 = 12
000C 253 :     FILHDR                 = 16
000C 254 :     STATBLK                = 20
000C 255 :     RTRVPTRLN             = 24
000C 256 :     RTRVPTRBUF            = 28
000C 257 :
000C 258 :     OFFSETS FROM FP
000C 259 :
000C 260 :     $OFFSET 0,NEGATIVE,<-
000C 261 :     <FID,6>,-
000C 262 :     <DIRNAM,16>,-
000C 263 :     <NAMBLK,10>,-
000C 264 :     <NAMDSC,12>,-
000C 265 :     <SCRATCHSIZE,0>-
000C 266 :     >
000C 267 :
000C 268 : THE FILE DESCRIPTION ON THE STACK LOOKS AS FOLLOWS:
000C 269 :
000C 270 :     +-----+
000C 271 :     | DIRECTORY NAME COUNT | :NAMDSC
000C 272 :     +-----+
000C 273 :     | ADDRESS OF DIRECTORY NAME |
000C 274 :     +-----+
000C 275 :     | ADDRESS OF NAMBLK |
000C 276 :     +-----+

```

00000000  
00000004  
00000008  
0000000C  
00000010  
00000014  
00000018  
0000001C

FFFA FID:  
FFEA DIRNAM:  
FFEO NAMBLK:  
FFD4 NAMDSC:  
FFD4 SCRATCHSIZE:



	000C	277	:		:	NAMBLK
	000C	278	:		:	
	000C	279	:		:	
	000C	280	:		:	
	000C	281	:		:	
	000C	282	:		:	
	000C	283	:		:	
	000C	284	:		:	DIRNAM
	000C	285	:		:	
	000C	286	:		:	
	000C	287	:		:	
	000C	288	:		:	
	000C	289	:		:	
	000C	290	:		:	
	000C	291	:		:	
	000C	292	:		:	
	000C	293	:		:	
	000C	294	:		:	
	000C	295	:		:	
	000C	296	:		:	
	000C	297	:		:	
	000C	298	:		:	
	000C	299	:		:	
	000C	300	:		:	
	000C	301	:		:	
	000E	313	:		:	
	0011	314	:		:	
	0016	315	:		:	
	001E	317	:		:	
	001E	318	:		:	
	001E	319	:		:	
	001E	320	:		:	
	001E	321	:		:	
	0023	322	:		:	
	0025	323	:		:	
	0025	324	:		:	
	0027	335	:		:	
	002E	336	:		:	
	0030	337	:		:	
	0033	338	:		:	
	0035	339	:		:	
	0037	340	:		:	
	003E	341	:		:	
	0041	342	:		:	
	0043	343	:		:	
	0046	344	:		:	
	0046	345	:		:	
	0046	346	:		:	
	0046	347	:		:	
	004A	348	:		:	
	004C	349	:		:	
	004F	350	:		:	
	0053	351	:		:	
	0055	352	:		:	
	0059	353	:		:	
	005C	354	:		:	
	005C	355	:		:	

  

04 BC	DC AD	5E	2C	08FC	000C	301	FIL\$OPENFILE::		
	00000000	EF			000E	313	.WORD	^M<R2,R3,R4,R5,R6,R7,R11>	
					0011	314	SUBL	#-SCRATCHSIZE,SP	:RESERVE SCRATCH STORAGE
					0016	315	MOVAL	NAMBLK(FP),NAMDESC+8(FP)	:SET ADDRESS OF NAME BLOCK
					001E	317	MOVL	BOO\$GL_RPBASE,@CHANADR(AP)	:INIT CHANNEL
					001E	318	:		
					001E	319	:	IF CACHE DESCRIPTOR EXISTS AND IS IN SYSTEM SPACE, THEN WE	
					001E	320	:	HAD BETTER BE IN KERNEL MODE TO USE THE CACHE.	
	5B	FFDE	CF	D0	001E	321	MOVL	W^FIL_GQ_CACHE,R11	:IS CACHE IN SYSTEM SPACE?
			02	14	0023	322	BGTR	10\$	:BRANCH IF DESCRIPTOR PRESENT
					0025	323			:AND NOT IN SYSTEM SPACE
	5B	00000004	EF	D0	0027	335	BEQL	20\$	:BRANCH IF NO DESCRIPTOR PRESENT
			07	13	002E	336	MOVL	FIL\$GQ_CACHE+4,R11	:IS THE CACHE ENABLED?
		01	68	B1	0030	337	BEQL	20\$	:BRANCH IF NOT
			02	13	0033	338	CMPW	FIL\$W_CACHE_ID(R11),#FIL\$C_CACHE_ID	:CORRECT VERSION OF CACHE?
			5B	D4	0035	339	BEQL	20\$	:BRANCH IF YES
	57	00000000	EF	DE	0037	340	CLRL	R11	:DISABLE THE CACHE
		56	87	9A	003E	341	MOVAL	FIL\$GT_DDSTRING,R7	:ADDRESS OF COUNTED STRING
			57	D6	0041	342	MOVZBL	(R7)+,R6	:GET BYTE COUNT
		56	02	C2	0043	343	INCL	R7	:STEP OVER BRACKET
					0046	344	SUBL	#2,R6	:DON'T COUNT THE BRACKETS
					0046	345	:		
					0046	346	:	GET FILE NAME STRING, AND STRIP DEVICE OFF IF PRESENT	
	50	08	AC	D0	0046	347	MOVL	FILNAM(AP),R0	:ADDRESS OF FILE NAME DESCRIPTOR
			27	13	004A	348	BEQL	32\$	:BRANCH IF NO NAME SPECIFIED
		52	60	7D	004C	349	MOVQ	(R0),R2	:R2 = SIZE, R3 = ADDRESS
	63	52	3A	3A	004F	350	LOCC	#^A/:/,R2,(R3)	:DEVICE NAME PRESENT?
			07	13	0053	351	BEQL	25\$	:BRANCH IF NOT
		53	01	A1	0055	352	MOVAB	1(R1),R3	:ADDRESS BEYOND ":"
			52	70	0059	353	MOVAB	-(R0),R2	:REMAINING SIZE
					005C	354	:		
					005C	355	:	SEE IF DIRECTORY SPECIFIED IN THE FILE NAME STRING	



```

63 5B 8F 91 005C 356 :
      05 13 005C 357 25$: CMPB #^A/[/, (R3) ; DIRECTORY DELIMITER?
63 3C 91 0060 358 BEQL 30$ ; BRANCH IF YES
      1D 12 0062 359 CMPB #^A/</, (R3) ; ALTERNATE CHARACTER
50 83 02 81 0065 360 BNEQ 40$ ; BRANCH IF NO DIRECTORY SPECIFIED
      52 D7 0067 361 30$: ADDB3 #2, (R3)+, R0 ; SCAN FOR MATCHING BRACKET ] OR >
63 52 50 3A 0068 362 DECL R2 ; ADJUST SIZE AND ADR OF STRING
      03 12 006D 363 LOCC R0, R2, (R3) ; SCAN FOR CLOSE BRACKET
      03 31 0071 364 BNEQ 35$ ; BRANCH IF FOUND IT
      03 31 0073 365 32$: BRW BADFILNAM ; BAD FILE NAME IF NO CLOSE BRACKET
56 57 53 D0 0076 366 35$: MOVL R3, R7 ; ADDRESS OF DIRECTORY NAME
      51 53 C3 0079 367 SUBL3 R3, R1, R6 ; SIZE OF DIRECTORY NAME
      52 70 9E 007D 368 MOVAB -(R0), R2 ; SIZE REMAINING SKIP CLOSE BRACKET
53 01 A1 DE 0080 369 MOVAL 1(R1), R3 ; ADR OF REMAINING STRING BEYOND CLOSE BRACKET
      0084 370 :
      0084 371 : SET UP COMMON ARGUMENT LIST FOR MOUNT, FINDFILID, RDCHKFILHDR
      0084 372 :
      0084 373 :
      0084 374 : ARGUMENT COUNT : AP
      0084 375 :
      0084 376 : CHANNEL NUMBER :
      0084 377 :
      0084 378 : NAME DESCRIPTOR :
      0084 379 :
      0084 380 : INDEX FILE HEADER BUF ADR :
      0084 381 :
      0084 382 : FILE HEADER BUFFER ADDR :
      0084 383 :
      0084 384 : ADDR OF STATISTICS BLOCK :
      0084 385 :
      0084 386 : ADDRESS OF FILE ID BLOCK :
      0084 387 :
      0084 388 : ADDR OF RTRV PTR LENGTH :
      0084 389 :
      0084 390 : ADDR OF RTRV PTR BUF DSCR :
      0084 391 :
      0084 392 :
      0084 393 40$: CLRQ -(SP) ; ASSUME NO RETRIEVAL POINTERS REQUESTED
      07 6C D1 0086 394 CMPL ARGCNT(AP), #RTRVPTRBUF/4 ; RETRIEVAL POINTER PARAMETERS PRESENT?
      04 19 0089 395 BLSS 45$ ; BRANCH IF NOT
6E 18 AC 7D 008B 396 MOVQ RTRVPTRLEN(AP), (SP) ; PUT RTRV PTR PARAMS IN LIST
      FA AD DF 008F 397 45$: PUSHAL FID(FP) ; ADDRESS OF FILE ID
7E 10 AC 7D 0092 398 MOVQ FILHDR(AP), -(SP) ; PUSH STATBLK ADR, FILHDR ADR
      0C AC DD 0096 399 PUSHL IXFHDR(AP) ; INDEX FILE HEADER ADDRESS
      D4 AD DF 0099 400 PUSHAL NAMDSC(FP) ; ADR OF 3 LONG WORD NAME DESCRIPTOR
      04 BC DD 009C 401 PUSHL @CHANADR(AP) ; CHANNEL TO USE, LONG WORD FOR BOOTING
      06 DD 009F 402 PUSHL #6 ; PARAMETER COUNT
      5B D5 00A1 403 TSTL R11 ; CACHE ENABLED?
      07 13 00A3 404 BEQL 50$
OC AE 18 AB DE 00A5 405 MOVAL FIL$A_IXFHDR(R11), IXFHDR(SP) ; USE CACHED INDEX FILE HEADER
      08 11 00AA 406 BRB 60$ ; AND SKIP THE MOUNT
      022C'CF 6E FA 00AC 407 50$: CALLG (SP), W^FIL$MOUNT ; 'MOUNT THE VOLUME' (READ HOME
      00B1 408 ; BLOCK, INDEX FILE HEADER, GET
      00B1 409 ; STRUCTURE LEVEL OF VOLUME)
      5D 50 E9 00B1 410 BLBC R0, 100$ ; BRANCH IF ERROR
      00B4 411 :
      00B4 412 : SET UP FOR THE DIRECTORY LOOK UP

```



51	FA AD 04	B0 00B4	413	:	MOVW	#FID\$C_MFD,FID(FP)	:MFD FILE NUMBER
	FC AD 04	D0 00B8	414	60\$:	MOVL	#FID\$C_MFD,FID+2(FP)	:MFD FILE SEQUENCE NO., RVN = 0
	FF48 CF	D5 00BC	415		TSTL	W^FIL_GT_TOPSYS	:TOP LEVEL SYSTEM DIRECTORY PRESENT?
	1A	13 00C0	416		BEQL	70\$	:BRANCH IF NOT
	00000000'EF	DE 00C2	417		MOVAL	FIL\$GT_TOPSYS,R1	:GET ADDRESS OF TOP LEVEL DIR STRING
	50 81	9A 00C9	418		MOVZBL	(R1)+,R0	:GET SIZE TO R0, ADR TO R1
	OE 13	00CC	419		BEQL	70\$	:BRANCH IF NONE SPECIFIED
	7E 56	7D 00CE	420		MOVQ	R6,-(SP)	:SAVE DIRECTORY STRING DESCRIPTOR
	56 50	7D 00D1	421		MOVQ	R0,R6	:TREAT TOPSYS LIKE DIR STRING
	00E9 30	00D4	422		BSBW	FORMDIRSTRING	:FORM THE DIRECTORY NAME
	56 8E	7D 00D7	423		MOVQ	(SP)+,R6	:RESTORE READ DIRECTORY DESCRIPTOR
	03 11	00DA	424		BRB	75\$	
	00E1 30	00DC	425	70\$:	BSBW	FORMDIRSTRING	:GET NEXT DIRECTORY TO LOOKUP
	D4 AD 50	7D 00DF	426	75\$:	MOVQ	R0,NAMDSC(FP)	:STORE DESCRIPTOR OF ITS NAME
	6E DF	00E3	427	80\$:	PUSHAL	(SP)	:REAL ADDRESS OF ARGUMENT LIST
	5B DD	00E5	428		PUSHL	R11	:CACHE ADDRESS IF ANY
	02B1'CF 02	FB 00E7	429		CALLS	#2,W^FIL\$FINDFILID	:FIND THE FILE ID
	22 50	E9 00EC	430		BLBC	R0,100\$	:BRANCH IF ERROR
	E0 AD 7C	00EF	431		CLRQ	NAMBLK(FP)	:REINIT NAME BLOCK
	E8 AD B4	00F2	432		CLRW	NAMBLK+8(FP)	
	56 D5	00F5	433		TSTL	R6	:ANY MORE DIRECTORY NAMES?
	E3 14	00F7	434		BGTR	70\$	:BRANCH IF YES, LOOKUP THE NEXT
	D4 AD 52	7D 00F9	435		MOVQ	R2,NAMDSC(FP)	:DESCRIPTOR FOR FILE TO LOOKUP
	52 D4	00FD	436		CLRL	R2	:STOP THE LOOKUP LOOP
	D4 AD D5	00FF	437		TSTL	NAMDSC(FP)	:ALREADY DONE?
	DF 14	0102	438		BGTR	80\$	:BRANCH IF NO, DO THE LAST ONE
	07 6C	D1 0104	439	85\$:	CMPL	ARGCNT(AP),#RTRVPTRBUF/4	:RETRIEVAL POINTERS DESIRED?
	03 19	0107	440		BLSS	90\$	:BRANCH IF NOT
	6E 02	C0 0109	441		ADDL	#2,(SP)	:ADDITIONAL ARGUMENTS ARE PRESENT
	0527'CF 6E	FA 010C	442	90\$:	CALLG	(SP),W^FIL\$RDCHKFILHDR	:READ AND CHECK FILE HEADER
		04 0111	443	100\$:	RET		
			444				
			445				
			446		.DSABL	LSB	



```

0112 448 .SBTTL FIL$CACHE_INIT - INIT FILEREAD CACHE
0112 449 :++
0112 450 : FUNCTIONAL DESCRIPTION:
0112 451 :
0112 452 :     CACHE_INIT PERFORMS THE INITIALIZATION FOR THE FILEREAD CACHE
0112 453 :
0112 454 : CALLING SEQUENCE:
0112 455 :
0112 456 :     CALLG  ARGLIST,FIL$CACHE_INIT
0112 457 :
0112 458 : INPUT PARAMETERS:
0112 459 :
0112 460 :     CHANADR(AP)      ADDRESS TO RETURN LONG WORD CHANNEL
0112 461 :     FILNAM(AP)      ADDRESS OF DEVICE NAME STRING DESCRIPTOR
0112 462 :                     THE DEVICE NAME MUST CONTAIN THE ":"
0112 463 :                     IF THE ADDRESS IS 0, THE STRING IS NULL,
0112 464 :                     OR THE NAME DOES NOT CONTAIN A ":", THE
0112 465 :                     DEFAULT DEVICE NAME IS USED
0112 466 :     CACHE_SIZE(AP)  SIZE IN BYTES OF FILEREAD CACHE
0112 467 :     CACHE_ADR(AP)  ADDRESS OF FILEREAD CACHE
0112 468 :     DIR_CACHE_CNT(AP) NUMBER OF DIRECTORY CACHE ENTRIES
0112 469 :     LBN_CACHE_CNT(AP) NUMBER OF LBN CACHE ENTRIES
0112 470 :
0112 471 : IMPLICIT INPUTS:
0112 472 :
0112 473 :     NONE
0112 474 :
0112 475 : OUTPUT PARAMETERS:
0112 476 :
0112 477 :     R0 = ALWAYS SUCCESSFUL STATUS CODE
0112 478 :
0112 479 : IMPLICIT OUTPUTS:
0112 480 :
0112 481 :     FIL$GQ_CACHE QUAD WORD FILLED IN WITH SIZE AND ADDRESS OF CACHE
0112 482 :
0112 483 : COMPLETION CODES:
0112 484 :
0112 485 :     SSS_NORMAL      SUCCESSFUL COMPLETION
0112 486 :
0112 487 : SIDE EFFECTS:
0112 488 :
0112 489 :     NONE
0112 490 :
0112 491 : EQUATED SYMBOLS, OFFSETS FROM AP
0112 492 :
0112 493 :
00000004 0112 494 :     CHANADR      =      4
00000008 0112 495 :     FILNAM       =      8
0000000C 0112 496 :     CACHE_SIZE   =     12
00000010 0112 497 :     CACHE_ADR    =     16
00000014 0112 498 :     DIR_CACHE_CNT =     20
00000018 0112 499 :     LBN_CACHE_CNT =     24
0112 500 :
0112 501 : --
0112 502 : FIL$CACHE_INIT::
0C3C 0112 503 :     .WORD  ^M<R2,R3,R4,R5,R10,R11>
0114 504

```

```

      0114 505
50 5A 5A 0C AC 7D 0114 506
      00000218 8F C3 0118 507
      6B 62 19 0120 508
      01 80 0122 509
      0125 510
      04 AB 00000218 8F D0 0125 511
      08 AB 00000218 8F D0 012D 512
      51 14 AC 24 C5 0135 513
      50 51 C2 013A 514
      45 19 013D 515
OC AB 51 00000218 8F C1 013F 516
      0148 517
      0148 518
      10 AB 0C AB D0 0148 519
      51 18 AC 09 78 014D 520
      50 51 D1 0152 521
      08 15 0155 522
51 50 000001FF 8F CB 0157 523
      14 AB 10 AB 51 C1 015F 524 20$:
      04 BC 00000000'EF D0 0165 529
      18 AB DF 016D 531
      7E D4 0170 532
      04 BC DD 0172 533
      022C'CF 03 FB 0175 534
      0A 50 E9 017A 535
      00000000'EF 5A 7D 017D 536
      50 01 D0 0184 537 100$:
      04 0187 538 110$:
      ASSUME CACHE_SIZE+4 EQ CACHE_ADR
      MOVQ CACHE_SIZE(AP),R10 ;R10=SIZE, R11=ADR
      SUBL3 #FILSC_SIZE,R10,R0 ;BYTES LEFT FOR DIR AND LBN CACHES
      BLSS 100$ ;BRANCH IF NOT ENOUGH CACHE SPACE
      MOVW #FILSC_CACHE_ID,FILSW_CACHE_ID(R11) ;SET CACHE ID
      ;ALLOWS MOVING CACHES BETWEEN FILEREAD'S
      MOVL #FILSC_SIZE,FILSL_DIROFF(R11) ;BEGINNING OF DIR CACHE
      MOVL #FILSC_SIZE,FILSL_DIRNXT(R11) ;NEXT AVAILABLE SLOT IN DIR CACHE
      MULL3 #FILSC_DIR_SIZE,DIR_CACHE_CNT(AP),R1 ;BYTE COUNT FOR DIR CACHE
      SUBL R1,R0 ;BYTE COUNT LEFT FOR LBN CACHE
      BLSS 100$ ;BRANCH IF NOT ENOUGH SPACE
      ADDL3 #FILSC_SIZE,R1,FILSL_DIRMAX(R11) ;END OF DIR CACHE
      ASSUME FILSL_DIRMAX EQ FILSL_LBNOFF
      MOVL FILSL_LBNOFF(R11),FILSL_LBNNXT(R11) ;NEXT LBN ENTRY TO ALLOCATE
      ASHL #9,LBN_CACHE_CNT(AP),R1 ;BYTE COUNT IN LBN CACHE
      CMPL R1,R0 ;ENOUGH ROOM FOR WHOLE LBN CACHE
      BLEQ 20$ ;BRANCH IF YES
      BICL3 #^X1FF,R0,R1 ;USE WHAT IS LEFT TRUNCATED
      ADDL3 R1,FILSL_LBNNXT(R11),FILSL_LBNMAX(R11) ;END OF LBN CACHE
      MOVL BOOSGL_RPBBASE,@CHANADR(AP);LOAD RPB BASE
      PUSHAL FILSA_IXFHDR(R11) ;ADDRESS TO READ INDEX FILE HEADER
      CLRL -(SP) ;UNUSED PARAMETER
      PUSHL @CHANADR(AP) ;CHANNEL JUST ASSIGNED
      CALLS #3,W^FILSMOUNT ;MOUNT THE VOLUME, RETURN INDEX FILE HDR
      BLBC R0,110$ ;BRANCH IF ERROR
      MOVQ R10,FILSGQ_CACHE ;SAVE DESCRIPTOR OF CACHE
      S^#SS$_NORMAL,R0
      RET
```



```

0188 540 .SBTTL FIL$CACHE_TRUNC - TRUNCATE FILEREAD CACHE
0188 541 :++
0188 542 : FUNCTIONAL DESCRIPTION:
0188 543 :
0188 544 :     CACHE_TRUNC TRUNCATES THE FILEREAD CACHE AND MAKES IT IMPOSSIBLE
0188 545 :     TO ADD MORE DIRECTORY CACHE OR DIRECTORY LBN ENTRIES TO IT. IN EFFECT
0188 546 :     THIS ROUTINE TURNS THE CACHE INTO A READ-ONLY DATA BASE.
0188 547 :
0188 548 : CALLING SEQUENCE:
0188 549 :
0188 550 :     CALLG  ARGLIST,FIL$CACHE_TRUNC
0188 551 :
0188 552 : INPUT PARAMETERS:
0188 553 :
0188 554 :     NONE
0188 555 :
0188 556 : IMPLICIT INPUTS:
0188 557 :
0188 558 :     FIL$GQ_CACHE          DESCRIPTOR FOR THE CACHE
0188 559 :
0188 560 : OUTPUT PARAMETERS:
0188 561 :
0188 562 :     RO = ALWAYS SUCCESSFUL STATUS CODE
0188 563 :
0188 564 : IMPLICIT OUTPUTS:
0188 565 :
0188 566 :     FIL$GQ_CACHE FILLED IN WITH ALTERED SIZE OF CACHE
0188 567 :
0188 568 : COMPLETION CODES:
0188 569 :
0188 570 :     SSS_NORMAL          SUCCESSFUL COMPLETION
0188 571 :
0188 572 : SIDE EFFECTS:
0188 573 :
0188 574 :     NONE
0188 575 :
0188 576 : EQUATED SYMBOLS
0188 577 :
0188 578 :
0188 579 :--
0188 580
0188 581 FIL$CACHE_TRUNC::
0188 582 .WORD 0
0188 583 MOVL  FIL$GQ_CACHE+4,RO          ;ADDRESS OF THE CACHE
0188 584 MOVL  FIL$DIRNXT(RO),FIL$DIRMAX(RO) ;NO NEW DIRECTORY CACHE ENTRIES
0188 585 MOVL  FIL$LBNNXT(RO),FIL$LBNNEXT(RO) ;NO MORE LBN BUFFERS
0188 586 MOVL  FIL$LBNNXT(RO),FIL$GQ_CACHE  ;SET NEW SIZE OF CACHE
0188 587 MOVL  S^#SSS_NORMAL,RO
0188 588 RET

```

```

50 00000004'EF 0000 DO 018A 583
OC AO 08 AO DO 0191 584
14 AO 10 AO DO 0196 585
00000000'EF 10 AO DO 019B 586
50 01 DO 01A3 587
04 01A6 588

```

```

01A7 636 .SBTTL STORE3DIGITS - STORE 3 ASCII DIGITS
01A7 637 :++
01A7 638 : FUNCTIONAL DESCRIPTION:
01A7 639 :
01A7 640 : STORE 3 DIGITS OF DIRECTORY STRING
01A7 641 :
01A7 642 : CALLING SEQUENCE:
01A7 643 :
01A7 644 : BSBB STORE3DIGITS
01A7 645 :
01A7 646 : INPUT:
01A7 647 :
01A7 648 : R0 = NO. OF DIGITS TO PUT IN STRING
01A7 649 : R1 = ADDRESS + 1 OF RIGHT MOST DIGIT
01A7 650 : R2 = ADDRESS AT WHICH TO STORE 3 DIGITS
01A7 651 :
01A7 652 : OUTPUTS:
01A7 653 :
01A7 654 : NONE
01A7 655 :
01A7 656 :--
01A7 657 :
01A7 658 STORE3DIGITS:
03 50 D1 01A7 659 CMPL R0,#3 ;3 DIGITS OR LESS SPECIFIED?
03 03 15 01AA 660 BLEQ 5$ ;YES. BRANCH.
0258 31 01AC 661 BRW BADFILNAM ;NO. DIRECTORY STRING BAD. EXIT
01AF 662 ;WITH ERROR.
82 3030 8F B0 01AF 663 5$: MOVW #A/00/,(R2)+ ;BACKGROUND WITH ASCII 0
82 30 90 01B4 664 MOVW #A/0/,(R2)+
03 11 01B7 665 BRB 20$ ;
72 71 90 01B9 666 10$: MOVW -(R1),-(R2) ;START LOOP AT BOTTOM
FA 50 F4 01BC 667 ;STORE BYTES LAST TO FIRST
05 01BF 668 20$: SOBGEQ R0,10$ ;LEAVING LEADING ASCII 0'S
01BF 669 RSB ;LOOP ZERO OR MORE TIMES

```



```

01C0 671 .SBTTL FORMDIRSTRING - GET A DIRECTORY STRING
01C0 672 :++
01C0 673 : FUNCTIONAL DESCRIPTION:
01C0 674 :
01C0 675 : PULL THE FIRST DIRECTORY NAME OFF THE FRONT OF THE INPUT
01C0 676 : DIRECTORY STRING AND FORM THE FULL FILE NAME OF THE DIRECTORY
01C0 677 : TO LOOK UP.
01C0 678 :
01C0 679 : CALLING SEQUENCE:
01C0 680 :
01C0 681 : BSBW FORMDIRSTRING
01C0 682 :
01C0 683 : INPUTS:
01C0 684 :
01C0 685 : R6 = SIZE OF DIRECTORY STRING
01C0 686 : R7 = ADDRESS OF DIRECTORY STRING
01C0 687 : THE STRING CONTAINS NO BRACKETS,
01C0 688 : IT MAY BE OF THE FORM "DIR1.DIR2.DIR3...DIRN"
01C0 689 : THE FIRST AND ONLY ITEM MAY BE IN THE FORM GROUP, MEMBER
01C0 690 : DIRNAM(FP) = ADDRESS OF AREA TO BUILD THE NAME
01C0 691 :
01C0 692 : OUTPUTS:
01C0 693 :
01C0 694 : R0 = SIZE OF DIRECTORY STRING
01C0 695 : R1 = ADDRESS OF DIRECTORY STRING
01C0 696 : R2,R3 PRESERVED
01C0 697 : R6,R7 UPDATED TO POINT AT THE REST OF THE STRING
01C0 698 :--
01C0 699 :
01C0 700 FORMDIRSTRING:
56 56 2E 3A 01C0 701 LOCC #^A/. /,R6,(R7) ;FIND NEXT DIRECTORY STRING
56 50 01 C3 01C4 702 SUBL3 #1,R0,R6 ;SIZE OF REST, SKIP THE ""
50 51 57 C3 01C8 703 ;-1 IF EMPTY
50 51 57 D0 01CC 704 SUBL3 R7,R1,R0 ;BYTE COUNT OF DIRECTORY NAME
57 01 A140 9E 01CF 705 MOVL R7,R1 ;ADDRESS OF DIRECTORY NAME
07 BB 01D4 706 MOVAB 1(R1)[R0],R7 ;ADDRESS OF NEXT BYTE BEYOND ""
09 50 D1 01D6 707 PUSHF #^M<R0,R1,R2> ;SAVE STRING DESCRIPTORS AND R2
03 15 01D9 708 CMPL R0,#9 ;LENGTH OF DIRECTORY STRING OKAY?
0229 31 01DB 709 BLEQ 5$ ;YES. BRANCH.
61 50 2C 3A 01DE 710 BRW BADFILNAM ;NO. EXIT WITH ERROR.
39 13 01E2 711 5$: LOCC #^A/. /,R0,(R1) ;GOOD STRING: ANY ""?
50 04 AE 50 C3 01E4 712 BEQL 20$ ;BRANCH IF NOT, RETURN THE DESCRIPTOR AS IS
52 EA AD DE 01EB 713 PUSHL R0 ;SAVE REMAINING BYTE COUNT
01 B6 10 01EF 714 SUBL3 R0,4(SP),R0 ;BYTE COUNT TO LEFT OF ""
50 8E 01 C3 01F1 715 MOVAL DIRNAM(FP),R2 ;ADDRESS TO STORE FIRST 3 CHARS
51 8E 8E C1 01F5 716 BSBB STORE3DIGITS ;STORE THEM
52 ED AD DE 01F9 717 SUBL3 #1,(SP)+,R0 ;COUNT OF CHARS TO RIGHT OF ""
A8 10 01FD 718 ADDL3 (SP)+,(SP)+,R1 ;ADR OF BYTE TO RIGHT OF LAST CHAR
04 BA 01FF 719 MOVAL DIRNAM+3(FP),R2 ;ADR TO STORE LAST 3 CHARS OF DIR NAME
50 06 D0 0201 720 BSBB STORE3DIGITS ;STORE THEM
51 EA AD40 9E 0204 721 POPR #^M<R2> ;RESTORE SAVED R2
81 5249442E 8F D0 0209 722 MOVL #6,R0 ;6 BYTES STRING SIZE
61 313B 8F B0 0210 723 10$: MOVAB DIRNAM(FP)[R0],R1 ;POINT TO END OF STRING
51 EA AD 9E 0215 724 MOVL #^A/.DIR/, (R1)+ ;PUT TYPE IN STRING
50 06 C0 0219 725 MOVW #^A/:1/, (R1) ;AND VERSION AS WELL
726 MOVAB DIRNAM(FP),R1 ;ADDRESS OF STRING
727 ADDL #6,R0 ;SIZE INCLUDES ".DIR;1"

```

			05	021C	728		RSB		
	38		BB	021D	729	20\$:	PUSHR	#^M<R3,R4,R5>	;SAVE THESE FROM MOV C3
				021F	730	:			
				021F	731	:			
				021F	732	:			
EA AD	10 BE	0C AE	28	021F	733		MOV C3	12(SP),@16(SP),DIRNAM(FP)	;MOVE NAME TO SCRATCH AREA
		38 BA	0226	734			POPR	#^M<R3,R4,R5>	;RESTORE REGISTERS
		07 BA	0228	735			POPR	#^M<R0,R1,R2>	
		D8 11	022A	736			BRB	10\$	



```

022C 738 .SBTTL MOUNT - MOUNT THE VOLUME, INIT FOR FILE LOOKUP
022C 739 :++
022C 740 : FUNCTIONAL DESCRIPTION:
022C 741 :
022C 742 : MOUNT PERFORMS THE NECESSARY INITIALIZATION FOR FILE LOOKUP.
022C 743 : IT READS THE HOME BLOCK, AND THEN RETURNS THE INDEX FILE HEADER TO THE
022C 744 : SPECIFIED BUFFER. THE INDEX FILE HEADER IS ALTERED BY RECORDING THE
022C 745 : VIRTUAL BLOCK OFFSET REQUIRED TO TRANSLATE 'FILE NUMBER' TO INDEX FILE VBN
022C 746 :
022C 747 : CALLING SEQUENCE:
022C 748 :
022C 749 : CALLG  ARGLIST,FIL$MOUNT
022C 750 :
022C 751 : INPUT PARAMETERS:
022C 752 :
022C 753 : CHAN(AP) CHANNEL ON WHICH DEVICE IS ASSIGNED
022C 754 : UNUSED 2ND PARAMETER NOT USED
022C 755 : IXFHDR(AP) ADDRESS TO RETURN INDEX FILE HEADER
022C 756 :
022C 757 : IMPLICIT INPUTS:
022C 758 :
022C 759 : NONE
022C 760 :
022C 761 : OUTPUT PARAMETERS:
022C 762 :
022C 763 : RO = SYSTEM STATUS CODE
022C 764 :
022C 765 : IMPLICIT OUTPUTS:
022C 766 :
022C 767 : NONE
022C 768 :
022C 769 : COMPLETION CODES:
022C 770 :
022C 771 : $$$_NORMAL SUCCESSFUL COMPLETION
022C 772 : $$$_FILESTRUCT FILE STRUCTURE LEVEL NOT SUPPORTED
022C 773 : $$$_BADCHKSUM CHECKSUM ERROR ON HOME BLOCK OR INDEX FILE HEADER
022C 774 : $$$_BADFILEHDR INDEX FILE HEADER IS BAD
022C 775 :
022C 776 : SIDE EFFECTS:
022C 777 :
022C 778 : NONE
022C 779 :
022C 780 :
022C 781 : EQUATED SYMBOLS, OFFSETS FROM AP
022C 782 :
022C 783 : CHAN = 4
022C 784 : IXFHDR = 12
022C 785 :
022C 786 : --
022C 787 :
022C 788 : FIL$MOUNT::
022C 789 : .WORD ^M<R2,R3,R4>
022E 790 : MOVL IXFHDR(AP),R3 ;ADDRESS OF BUFFER
0232 791 : ROTL #9,#1,-(SP) ;NUMBER OF BYTES TO READ
0236 792 : MOVZWL #10$_READLBLK,-(SP) ;READ LOGICAL BLOCK FUNCTION
0239 793 : PUSHL R3 ;BUFFER ADDRESS
023B 794 : PUSHL #1 ;LOGICAL BLOCK NUMBER 1 IS HOME BLK

```

00000004  
0000000C

53 0C AC 001C  
7E 01 09 9C  
7E 21 3C  
53 DD  
01 DD

```

04 AC DD 023D 795 PUSHL CHAN(AP) ;CHANNEL
05 DD 0240 796 PUSHL #5 ;NO. OF ARGUMENTS
0000'CF 6E FA 0242 797 CALLG (SP),W^FIL$RDWRTLBN ;READ THE HOME BLOCK
60 50 E9 0247 798 BLBC R0,30$ ;BRANCH IF ERROR
51 53 D0 024A 799 MOVL R3,R1 ;ADDRESS OF HOME BLOCK
50 1D 3C 024D 800 MOVZWL #HM2$W_CHECKSUM1@-1,R0 ;NO. OF WORDS IN FIRST CHECKSUM
04CB 30 0250 801 BSBW FIL$CHECKSUM1 ;CHECK THE FIRST CHECKSUM
51 53 D0 0253 802 MOVL R3,R1 ;ADR OF HOME BLOCK AGAIN
04C0 30 0256 803 BSBW FIL$CHECKSUM ;CHECK THE MAIN CHECKSUM
02 OD A3 91 0259 804 CMPB HM2$B_STRUCLEV(R3),#2 ;IS THIS STRUCTURE LEVEL 2?
4C 12 025D 805 BNEQ 40$ ;BR IF NOT STRUCTURE LEVEL 2
025F 806
025F 807 ; STRUCTURE LEVEL 2
025F 808
51 18 A3 D0 025F 809 MOVL HM2$L_IBMAPLBN(R3),R1 ;INDEX BIT MAP STARTING LBN
50 20 A3 3C 0263 810 MOVZWL HM2$W_IBMAPSIZE(R3),R0 ;INDEX BIT MAP SIZE IN BLOCKS
54 1C A3 D0 0267 811 MOVL HM2$L_MAXFILES(R3),R4 ;MAXIMUM FILES ON VOLUME
026B 812 ;ONLY INTERESETED IN HIGH 16 BITS
54 OE A3 04 A5 026B 813 MULW3 #4,HM2$W_CLUSTER(R3),R4 ;4*CLUSTER TO LOW WORD OF R4
54 OE A3 04 A5 0270 814 MULW3 #4,HM2$W_CLUSTER(R3),R4 ;4*CLUSTER TO LOW WORD OF R4
54 54 50 A0 0275 815 ADDW R0,R4 ;LOW WORD IS VBNOFFSET
0278 816 ;FROM FILE ID TO INDEX FILE VBN
0278 817
0278 818 ; READ INDEX FILE HEADER
0278 819 R0 - NUMBER OF BLOCKS IN INDEX FILE
0278 820 R1 - STARTING LBN OF INDEX FILE
0278 821
08 AE 51 50 C1 0278 822 ADDL3 R0,R1,8(SP) ;DESIRED LBN TO ARG LIST
0000'CF 8E FB 027D 823 CALLS (SP)+,W^FIL$RDWRTLBN ;READ INDEX FILE HEADER
0282 824 ;STRIP OFF THE ARGUMENT LIST
25 50 E9 0282 825 BLBC R0,30$ ;BRANCH IF ERROR
51 53 D0 0285 826 MOVL R3,R1 ;ADDRESS OF HEADER
7E D4 0288 827 CLRL -(SP) ;FORM FILE ID ON STACK
00010001 8F DD 028A 828 PUSHL #^X10001 ;FOR THE INDEX FILE HEADER
50 5E D0 0290 829 MOVL SP,R0 ;ADDRESS OF FILE ID
0453 30 0293 830 BSBW FIL$CHKFILHDR ;CHECK THE FILE HEADER (SEE IF
01FE C3 54 B0 0296 831 ;FILE IDS MATCH)
0296 832 MOVW R4,FH2$W_VBNOFFSET(R3) ;STORE VBN OFFSET
0298 833
0298 834 ; IF MAXFILES WAS GREATER THAN ^XFFFF THEN HIGH 16 BITS OF R4 WILL BE
0298 835 ; NON-ZERO. IN THIS CASE, RECORD THE BIGFILNUM BIT IN THE STRUCLEV WORD
0298 836
54 54 F0 8F 78 0298 837 ASHL #-16,R4,R4 ;SEE IF HIGH 16 BITS = 0
05 13 02A0 838 BEQL 25$
00 06 A3 0A E2 02A2 839 BBSS #FH2$V_BIGFILNUM,FH2$W_STRUCLEV(R3),25$ ;MUST USE HIGH 8 BITS
02A7 840 ;OF RVN FIELD AS FILE NUMBER EXTENSION
50 01 3C 02A7 841 25$: MOVZWL #SS$_NORMAL,R0 ;SUCCESSFUL COMPLETION
04 02AA 842 30$: RET
50 08C0 8F 3C 02AB 843 40$: MOVZWL #SS$_FILESTRUCT,R0 ;UNSUPPORTED FILE STRUCTURE LEVEL
04 02B0 844 RET

```



```

0281 846 .SBTTL FINDFILID - FIND FILE ID FOR SPECIFIED FILE
0281 847 :++
0281 848 : FUNCTIONAL DESCRIPTION:
0281 849 :
0281 850 : FINDFILID SCANS A SPECIFIED DIRECTORY FOR A FILE AND
0281 851 : RETURNS ITS FILE ID IF FOUND. STRUCTURE LEVEL 2 DIRECTORIES
0281 852 : ARE SUPPORTED, 0 VERSION NUMBER MEANS FIND MOST RECENT VERSION,
0281 853 : -1 VERSION (FIND OLDEST) IS NOT SUPPORTED.
0281 854 :
0281 855 : NOTE THAT NON-CONTIGUOUS DIRECTORIES ARE NOT SUPPORTED.
0281 856 :
0281 857 : CALLING SEQUENCE:
0281 858 :
0281 859 : CALLG  ARGLIST,FIL$FINDFILID
0281 860 :
0281 861 : INPUT PARAMETERS:
0281 862 :
0281 863 :     CHAN(AP)      =      :CHANNEL ON WHICH DEVICE IS ASSIGNED
0281 864 :     FILDSC(AP)    =      ADDRESS OF 3 LONG WORD FILE DESCRIPTOR
0281 865 :                        1 - SIZE OF ASCII STRING, A 0 VALUE MEANS
0281 866 :                        USE THE CONTENTS OF THE NAMBLK BELOW
0281 867 :                        2 - ADDRESS OF ASCII STRING
0281 868 :                        3 - ADDRESS OF NAME BLOCK - (OBSOLETE, LEVEL 1 ONLY)
0281 869 :                        MAY CONTAIN DEFAULTS, BUT MUST BE
0281 870 :                        AT LEAST INITIALIZED TO ZERO
0281 871 :                        IT WILL BE WRITTEN.
0281 872 :     IXFHDR(AP)    ADR OF INDEX FILE HDR AS RETURNED FROM FIL$MOUNT
0281 873 :     DIRBUF(AP)    ADR OF 512 BYTE BUFFER TO USE FOR DIRECTORY SCAN
0281 874 :     STATBLK(AP)   ADDRESS OF 2 LONG WORD AREA USED FOR A
0281 875 :                   SCRATCH STATISTICS BLOCK
0281 876 :     FILID(AP)     ADR OF 3 WORD AREA USED BOTH AS THE ID OF
0281 877 :                   THE DIRECTORY TO SCAN AND AS THE PLACE TO
0281 878 :                   RETURN THE ID OF THE FILE FOUND
0281 879 :
0281 880 : IMPLICIT INPUTS:
0281 881 :
0281 882 :     NONE
0281 883 :
0281 884 : OUTPUT PARAMETERS:
0281 885 :
0281 886 :     RO = SYSTEM STATUS CODE
0281 887 :
0281 888 : IMPLICIT OUTPUTS:
0281 889 :
0281 890 :     NONE
0281 891 :
0281 892 : COMPLETION CODES:
0281 893 :
0281 894 :     $$$_NORMAL      SUCCESSFUL COMPLETION
0281 895 :     $$$_NOSUCHFILE  FILE NOT FOUND
0281 896 :     $$$_BADFILENAME SYNTAX ERROR IN FILE NAME
0281 897 :     $$$_BADCHKSUM   CHECKSUM ERROR ON DIRECTORY FILE HEADER
0281 898 :     $$$_BADFILEHDR  DIRECTORY FILE HEADER WAS BAD
0281 899 :
0281 900 : SIDE EFFECTS:
0281 901 :
0281 902 :     NONE

```



```
02B1 903 :
02B1 904 :
02B1 905 : EQUATED SYMBOLS, OFFSETS FROM AP
02B1 906 :
00000004 02B1 907 CHAN = 4
00000008 02B1 908 FILDSC = 8
0000000C 02B1 909 IXFHDR = 12
00000010 02B1 910 DIRBUF = 16
00000014 02B1 911 STATBLK = 20
00000018 02B1 912 FILID = 24
02B1 913 :
02B1 914 : OFFSETS FROM FP
02B1 915 :
02B1 916 $OFFSET 0,NEGATIVE,<-
02B1 917 DIR_BFCNT,- :BUFFER COUNT REMAINING IN LBN CACHE
02B1 918 DIR_BUF,- :NEXT BUFFER ADDRESS IN DIR LBN CACHE
02B1 919 ENTRY_ADR,- :FOUND CACHE ENTRY ADR
02B1 920 <ENTRY,FIL$C DIR_SIZE>,- :CACHE ENTRY FOR SEARCH/CREATE
02B1 921 <SCRATCH_SIZE,0>=- :SIZE OF SCRATCH AREA
02B1 922 >
FFFC DIR_BFCNT:
FFF8 DIR_BUF:
FFF4 ENTRY_ADR:
FFD0 ENTRY:
FFD0 SCRATCH_SIZE:
02B1 923 :
02B1 924 :--
02B1 925 :
02B1 926 FIL$FINDFILID::
02B1 927 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
02B3 928 SUBL #-SCRATCH_SIZE,SP :ALLOCATE SCRATCH SPACE
02B6 929 MOVCS #0,(SP),#0,#-SCRATCH_SIZE,(SP) :ZERO THE SCRATCH STORAGE
02BC 930 CLRL R11 :ASSUME NO CACHE
02BE 931 CMPL (AP),#2 :IF ONLY 2 ARGUMENTS
02C1 932 BNEQ 5$
02C3 933 MOVL 4(AP),R11 :THE FIRST IS THE CACHE ADDRESS
02C7 934 MOVL 8(AP),AP :THE SECOND IS THE REAL ARGUMENT LIST
02CB 935 5$: TSTL R11 :CACHE ENABLED?
02CD 936 BEQL 65$ :BRANCH IF NOT
02CF 937 :
02CF 938 : WE DO HAVE A CACHE TO LOOK IN AND MAKE ENTRIES IN, SET UP THE
02CF 939 : SCRATCH REGION FOR A LOOKUP
02CF 940 :
02CF 941 ASSUME ENTRY EQ SCRATCH_SIZE :SCRATCH CACHE ENTRY MUST BE ON TOP
02CF 942 ASSUME FIL$A DIR_FID EQ 0 :DIR ID IS AT FRONT OF CACHE ENTRY
02CF 943 MOVCS #6,@FILID(AP),(SP) :STORE DIRECTORY ID
02D4 944 MOVQ @FILDSC(AP),R0 :GET LOOKUP NAME DESCRIPTOR
02D8 945 CMPL R0,#6 :MUST BE MORE THAN ".DIR;1"
02DB 946 BLEQ 10$ :BRANCH IF NOT A DIRECTORY NAME
02DD 947 CMPL R0,#15 :AT MOST 9 CHAR WITH ".DIR;1"
02E0 948 BGTR 10$ :BRANCH IF NOT A DIRECTORY NAME
02E2 949 ADDL3 R0,R1,R2 :POINT OFF END OF NAME STRING
02E6 950 CMPW #^A/;1/,-(R2) :LAST 2 BYTES ".1" ?
02EB 951 BNEQ 10$ :BRANCH IF NOT A DIRECTORY NAME
02ED 952 CMPL #^A/.DIR/,-(R2) :PRECEDED BY ".DIR" ?
02F4 953 BNEQ 10$ :BRANCH IF NOT A DIRECTORY NAME
02F6 954 SUBL #6,R0 :JUST KEEP THE NAME PART
```



```
D7 AD D6 AD 50 90 02F9 955      MOVB      R0,FILST_DIR_NAM+ENTRY(FP) ;PUT SIZE AND NAME STRING
58 5B 61 50 28 02FD 956      MOVCL3     R0,(R1),FILST_DIR_NAM+1+ENTRY(FP) ;IN ENTRY TO LOOKUP
59 5B 04 AB C1 0302 957 10$: ADDL3      FILSL_DIROFF(R11),R11,R8 ;ADDRESS OF DIRECTORY CACHE
                                ADDL3      FILSL_DIRNXT(R11),R11,R9 ;ADDRESS OF LAST+1 BYTE
                                BRB        60$ ;LOOP 0 OR MORE TIMES
                                030E      960
                                030E      961
68 D0 AD 10 29 030E 962 20$: ASSUME     FILST_DIR_NAM EQ FILSA_DIR_FID+6
                                CMPC3      #6+10,FILSA_DIR_FID+ENTRY(FP), - ;DOES FID AND NAME MATCH?
                                0313      963
                                0313      964      BNEQ      30$ ;BRANCH IF DIDN'T MATCH ALL OF IT
                                F4 AD 58 D0 0315 965      MOVL      R8,ENTRY_ADR(FP) ;RECORD THAT A MATCH WAS FOUND
                                1E 11 0319 966      BRB        70$
                                031B      967
                                031B      968 ; FAILED TO MATCH THE ENTIRE ENTRY, DID WE MATCH THE DIR ID FIELD?
                                031B      969
                                031B      970 30$: CMPL      R0,#10 ;IF 10 OR LESS CHAR'S LEFT
                                031E      971      BGTR      50$ ;THEN MATCHED THE DIR ID
                                F4 AD 58 D0 0320 972      MOVL      R8,ENTRY_ADR(FP) ;SAVE THIS PARTIAL MATCH
                                D6 AD 95 0324 973      TSTB      FILST_DIR_NAM+ENTRY(FP) ;IF NOT SEARCHING FOR A DIR NAME
                                10 13 0327 974      BEQL      70$ ;THEN THIS ENTRY WILL DO FINE
                                58 24 C0 0329 975 50$: ADDL      #FILSC_DIR_SIZE,R8 ;ADDRESS OF NEXT CACHE ENTRY
                                59 58 D1 032C 976 60$: CMPL      R8,R9 ;DONE SCANNING DIR CACHE?
                                DD 1F 032F 977      BLSSU     20$ ;BRANCH IF NOT, CHECK NEXT ENTRY
                                0331      978
                                0331      979 ; ENTRY_ADR(FP) = ADDRESS OF CACHE HIT ENTRY
                                0331      980 ; = 0 IF NO MATCH FOUND
                                0331      981 ; IF WE DROP THROUGH TO HERE AND WE GOT A CACHE HIT, THEN IT WAS
                                0331      982 ; NOT EXACTLY WHAT WE WERE LOOKING FOR. BUT IT DID MATCH THE DIRECTORY.
                                0331      983
                                58 F4 AD D0 0331 984      MOVL      ENTRY_ADR(FP),R8 ;WAS THERE A CACHE HIT?
                                3E 13 0335 985 65$: BEQL      READ_DIR_HEADER ;BRANCH IF NO
                                OF 11 0337 986      BRB        80$ ;YES, FOR DIRECTORY LBN AND SIZE
                                0339      987
                                0339      988 ; FOUND WHAT WE WERE LOOKING FOR - MAY ONLY NEED DIRECTORY LBN AND SIZE
                                0339      989
                                1E A8 D5 0339 990 70$: TSTL      FILSA_DIR_OFID(R8) ;DID WE GET A FILE ID?
                                0A 13 033C 991      BEQL      80$ ;BRANCH IF NOT
                                18 BC 1E A8 06 28 033E 992      MOVCL3     #6,FILSA_DIR_OFID(R8),AFILID(AP) ;RETURN THE FILE ID
                                50 01 3C 0344 993      MOVZWL     S^#SS$_NORMAC,R0 ;SET SUCCESS STATUS
                                04 0347 994 72$: RET ;AND RETURN
                                0348      995
                                0348      996 ; CACHE HIT ONLY FOUND THE DIRECTORY LBN AND SIZE, SAVING THE
                                0348      997 ; READ OF THE DIRECTORY FILE HEADER.
                                0348      998
                                E0 AD 10 A8 7D 0348 999 80$: MOVQ      FILSQ_DIR_HDR(R8),FILSQ_DIR_HDR+ENTRY(FP) ;SAVE DIRHDR
                                034D 1000 ;INFO FOR MAKING A NEW ENTRY
                                034D 1001 ;WITH THE DIRECTORY FID IN IT
                                E8 AD 18 A8 D0 034D 1002      MOVL      FILSL_DIR_BFOFF(R8),FILSL_DIR_BFOFF+ENTRY(FP)
                                EC AD 1C A8 B0 0352 1003      MOVW      FILSW_DIR_BFCNT(R8),FILSW_DIR_BFCNT+ENTRY(FP)
                                0357 1004 ;SAVE DIR LBN CACHE INFO TOO
                                0357 1005
                                0357 1006 ; AT THIS POINT ENTRY(FP) CONTAINS DIRECTORY LBN CACHE INFORMATION
                                0357 1007 ; IF ONE HAD ALREADY EXISTED OR IF WE JUST CREATED IT.
                                0357 1008 ; SET UP THE WORKING LOCATIONS FOR THE DIRECTORY LBN CACHE
                                0357 1009
                                56 E4 AD 01 C3 0357 1010 90$: SUBL3     #1,FILSL_DIR_LBN+ENTRY(FP),R6 ;R6=STARTING LBN - 1
                                57 E0 AD 3C 035C 1011      MOVZWL     FILSW_DIR_BKCNT+ENTRY(FP),R7 ;R7=SIZE OF DIRECTORY IN BLOCKS
```



```

FC AD EC AD 3C 0360 1012 MOVZWL FIL$W_DIR_BFCNT+ENTRY(FP),DIR_BFCNT(FP) ;BUFFER COUNT IN LBN CACHE
F8 AD 5B E8 AD C1 0365 1013 ADDL3 FIL$L_DIR_BFOFF+ENTRY(FP),R11,DIR_BUF(FP) ;STARTING ADR IN CACHE
57 56 C0 036B 1014 ADDL R6,R7 ;LAST LBN OF FILE INCLUSIVE
00A8 31 036E 1015 BRW FIND_LEVEL2_1
008D 31 0371 1016 BADDIR2: BRW BADDIR
0371 1017 BADRET1: RET
0374 1018
04 0374 1019
0375 1020
0375 1021 : CACHE WAS NOT ENABLED OR THERE WAS NOT A HIT FOR THIS DIRECTORY
0375 1022
0375 1023 READ_DIR_HEADER:
0375 1024 CALLG (AP),FIL$RDCHKFILHDR ;READ AND CHECK DIRECTORY FILE HEADER
F5 50 E9 037C 1025 BLBC R0,BADRET1 ;BRANCH IF ERROR
55 10 AC D0 037F 1026 MOVL DIRBUF(AP),R5 ;ADDRESS OF BUFFER TO READ DIRECTORY BLOCKS
56 14 BC 01 C3 0383 1027 SUBL3 #1,@STATBLK(AP),R6 ;GET START LBN - 1
0388 1028
0388 1029 : IF THIS RESULT IS NEGATIVE, THEN THE DIRECTORY WAS NOT CONTIGUOUS.
0388 1030 : THIS CODE SUPPORTS ONLY CONTIGUOUS DIRECTORIES, ANOTHER BUFFER WOULD
0388 1031 : BE REQUIRED TO HOLD THE DIRECTORY HEADER IN ORDER TO SUPPORT NON-CONTIGUOS
0388 1032 : DIRECTORIES. SUCH DIRECTORIES ARE ONLY CREATED BY FILES-11 WHEN
0388 1033 : A DIRECTORY MUST BE EXTENDED AND THERE IS NOT ENOUGH CONTIGUOUS SPACE
0388 1034 : ANYWHERE ON THE VOLUME TO MAKE A NEW DIRECTORY OF THE CORRECT SIZE.
0388 1035
E7 19 0388 1036 BLSS BADDIR2 ;BRANCH IF NOT CONTIGUOUS
038A 1037
038A 1038 : SEE IF THIS LOOKS LIKE A DIRECTORY FILE
038A 1039
54 14 A5 DE 038A 1040 MOVAL FH2$W_RECATTR(R5),R4 ;ADDRESS OF LEVEL 2 RECORD ATTRIBUTES
57 08 A4 10 9C 038E 1041 10$: ROTL #16,FAT$L_EFBLK(R4),R7 ;VBN OF DIRECTORY EOF
OC A4 B5 0393 1042 TSTW FAT$W_FFBYTE(R4) ;IF ZERO, EFBLK IS LAST+1 VBN
02 12 0396 1043 BNEQ 20$
57 D7 0398 1044 DECL R7 ;CORRECT TO GET LAST VBN
E4 AD 56 01 C1 039A 1045 20$: ADDL3 #1,R6,FIL$L_DIR_LBN+ENTRY(FP) ;SAVE START LBN,
EO AD 57 B0 039F 1046 MOVW R7,FIL$W_DIR_BKCNT+ENTRY(FP) ;DIRECTORY SIZE,
E2 AD 01 90 03A3 1047 MOVB #1,FIL$B_DIR_LVL+ENTRY(FP) ;AND INDIC ODS-2 STRUCTURE LEVEL
03A7 1048 ;(THIS IS FOR BACKWARD-COMPAT W/SYSBOOT)
03A7 1049
03A7 1050 : SEE IF WE CAN SET UP A CACHE OF THE DIRECTORY BLOCKS FOR THIS DIRECTORY
03A7 1051
5B D5 03A7 1052 TSTL R11 ;ANY CACHEING ENABLED?
51 13 03A9 1053 BEQL 80$ ;BRANCH IF NOT
52 14 AB 10 AB C3 03AB 1054 SUBL3 FIL$L_LBNNXT(R11),FIL$L_LBNMAX(R11),R2 ;NO. OF BYTES
03B1 1055 ;AVAILABLE TO ALLOCATE
52 52 F7 8F 78 03B1 1056 ASHL #-9,R2,R2 ;NO. OF PAGES AVAILABLE
44 13 03B6 1057 BEQL 80$ ;BRANCH IF NO SPACE AT ALL
57 52 D1 03B8 1058 CMPL R2,R7 ;ENOUGH ROOM FOR WHOLE DIR
03 15 03BB 1059 BLEQ 40$ ;BRANCH IF NOT, USE WHAT IS LEFT
52 57 D0 03BD 1060 MOVL R7,R2 ;YES, USE THE RIGHT SIZE
53 10 AB D0 03C0 1061 40$: MOVL FIL$L_LBNNXT(R11),R3 ;OFFSET TO DIR LBN CACHE
03C4 1062
03C4 1063 : READ THE DISK BLOCKS INTO THE LBN CACHE.
03C4 1064
7E 52 09 78 03C4 1065 ASHL #9,R2,-(SP) ;BYTE COUNT TO TRANSFER
7E 21 3C 03C8 1066 MOVZWL #10$ READLBLK,-(SP) ;READ LOGICAL BLOCK FUNCTION
6843 9F 03CB 1067 PUSHAB (R11)[R3] ;BUFFER ADDRESS
E4 AD DD 03CE 1068 PUSHL FIL$L_DIR_LBN+ENTRY(FP) ;STARTING LBN

```



```

00000000'EF 04 AC DD 03D1 1069      PUSHL  CHAN(AP)      ;CHANNEL
              05 FB 03D4 1070      CALLS   #5,FIL$RDWRTLBN ;FILL THE DIR LBN CACHE
              1E 50 E9 03DB 1071      BLBC    R0,80$      ;BRANCH IF ERROR
              03DE 1072      :
              03DE 1073      : NOTE THAT DIRECTORY BLOCKS ARE IN MEMORY
              03DE 1074      :
              FC AD 52 DO 03DE 1075      MOVL    R2,DIR_BFCNT(FP) ;COUNT OF BLOCKS READ IN
              F8 AD 6B43 9E 03E2 1076      MOVAB   (R11)[R3],DIR_BUF(FP) ;ADDRESS OF FIRST BLOCK READ IN
              D6 AD 95 03E7 1077      TSTB    FIL$T_DIR_NAM+ENTRY(FP) ;ARE WE LOOKING UP ANOTHER DIRECTORY?
              10 12 03EA 1078      BNEQ     80$      ;BRANCH IF YES, DON'T ALLOCATE
              03EC 1079      : ;A PERMANENT CACHE ENTRY FOR
              03EC 1080      : ;AN INTERMEDIATE LEVEL DIRECTORY
              EC AD 52 B0 03EC 1081      MOVW    R2,FIL$W_DIR_BFCNT+ENTRY(FP) ;SET UP FOR PERMANENT ENTRY
              E8 AD 53 D0 03F0 1082      MOVL    R3,FIL$L_DIR_BFOFF+ENTRY(FP) ;SAVE THE SIZE AND OFFSET OF CACHE
              51 52 09 78 03F4 1083      ASHL    #9,R2,R1 ;NO. OF BYTES IN CACHE
              10 AB 51 C0 03F8 1084      ADDL    R1,FIL$L_LBNXNT(R11) ;ALLOCATE THE CACHE
              03FC 1085      :
              03FC 1086      : IF IT WAS POSSIBLE TO READ THE DIRECTORY INTO THE LBN CACHE AREA,
              03FC 1087      : THE SCAN WILL FIND THESE BLOCKS AS THEY ARE NEEDED.
              03FC 1088      :
              03FC 1089      80$:
              03FC 1090      :
              03FC 1091      : R6 = STARTING LBN - 1 FOR THE DIRECTORY
              03FC 1092      : R7 = COUNT OF BLOCKS OF DIRECTORY TO BE SCANNED
              03FC 1093      :
              57 56 C0 03FC 1094      ADDL    R6,R7 ;LAST LBN OF FILE (INCLUSIVE)
              OC 11 03FF 1095      BRB FIND_LEVEL2
              0401 1096      :
              0401 1097      :
              0401 1098      : ERROR RETURNS.
              0401 1099      :
              50 0828 8F 3C 0401 1100      BADDIR: MOVZWL #SS$_BADIRECTORY,R0
              04 0406 1101      BADRET: RET
              0407 1102      BADFILNAM:
              50 0818 8F 3C 0407 1103      MOVZWL #SS$_BADFILENAME,R0 ;RETURN ERROR CODE.
              04 040C 1104      RET

```



```

040D 1106 .SBTTL FIL$FINDFILID - STRUCTURE LEVEL 2
040D 1107 :
040D 1108 : STRUCTURE LEVEL 2
040D 1109 :
040D 1110 FIND_LEVEL2:
EF 34 A5 OD E1 040D 1111 BBC #FH2$V_DIRECTORY,FH2$L_FILECHAR(R5),BADDIR ;DIRECTORY BIT MUST BE SE
0412 1112
0412 1113 ASSUME FATS$B_RATTRIB EQ FATS$B_RTYPE+1
0802 8F B1 0412 1114 CMPW #<FATS$M_NOSPAN @ 8 + FATS$C_VARIABLE>,- ;VARIABLE LENGTH
64 0416 1115 FATS$B_RTYPE(R4) ;RECORDS NOT CROSSING BLOCK BOUNDARIES
E8 12 0417 1116 BNEQ BADDIR ;BRANCH IF BAD RECORD ATTRIBUTES
0419 1117 :
0419 1118 : ***** NOTE THAT EACH BLOCK MUST END IN A RECORD SIZE OF -1
0419 1119 : ***** A RECORD IS NOT ALLOWED TO EXACTLY FILL THE BLOCK
0419 1120 : ***** PDP-11 FILE CONTROL SERVICES WILL READ THIS FILE CORRECTLY, BUT
0419 1121 : ***** WILL NOT WRITE IT PROPERLY. LIKEWISE FOR RMS-11 AND RMS-32
0419 1122 :
0419 1123 FIND_LEVEL2 1:
58 08 BC 7D 0419 1124 MOVQ @FILDSC(AP),R8 ;R8 = SIZE, R9 = ADDRESS OF FILE NAME STRING
5A D4 041D 1125 CLRL R10 ;ASSUME DEFAULT VERSION
64 53 58 7D 041F 1126 MOVQ R8,R3 ;COPY FILE NAME DESCRIPTOR
64 53 2E 3A 0422 1127 LOCC #^A/./,R3,(R4) ;FIND FILE TYPE DELIMITER IF PRESENT
07 13 0426 1128 BEQL 40$ ;BRANCH IF NOT PRESENT
53 70 9E 0428 1129 MOVAB -(R0),R3 ;SIZE OF REMAINING STRING
54 01 A1 9E 042B 1130 MOVAB 1(R1),R4 ;ADDRESS OF STRING BEYOND DELIMITER
64 53 3B 3A 042F 1131 40$: LOCC #^A/./,R3,(R4) ;SEE IF VERSION DELIMITER PRESENT
06 12 0433 1132 BNEQ 60$ ;BRANCH IF IT IS
64 53 2E 3A 0435 1133 LOCC #^A/./,R3,(R4) ;TRY ALTERNATE VERSION DELIMITER
44 13 0439 1134 BEQL 120$ ;BRANCH IF NO VERSION STRING PRESENT
043B 1135 :
043B 1136 : R0 = BYTE COUNT OF VERSION STRING PLUS DELIMITER
043B 1137 : R1 = ADDRESS OF VERSION DELIMITER
043B 1138 :
58 50 C2 043B 1139 60$: SUBL R0,R8 ;REDUCE FILE NAME STRING SIZE
043E 1140 ;ELIMINATING VERSION STRING AND DELIMITER
7E DF 043E 1141 PUSHAL -(SP) ;RESERVE LONG WORD FOR VERSION NUMBER
0440 1142 ;AND PUSH ITS ADDRESS
01 A1 9F 0440 1143 PUSHAB 1(R1) ;ADDRESS OF VERSION STRING
70 9F 0443 1144 PUSHAB -(R0) ;SIZE OF VERSION STRING
00000000'EF 03 FB 0445 1145 CALLS #3,LIB$CVT_DTB ;CONVERT DECIMAL VERSION STRING TO BINARY
B8 50 E9 044C 1146 BLBC R0,BADFILNAM ;BRANCH IF SYNTAX ERROR IN VERSION STRING
5A 8E D0 044F 1147 MOVL (SP)+,R10 ;FETCH EXPLICIT VERSION NUMBER
0452 1148 :
0452 1149 : R6 = ADDRESS OF LAST LBN READ FROM DIRECTORY FILE (FIRST - 1)
0452 1150 : R7 = ADDRESS OF LAST LBN (INCLUSIVE) TO BE READ FROM DIRECTORY FILE
0452 1151 : R8 = SIZE OF NAME STRING TO SCAN FOR
0452 1152 : R9 = ADDRESS OF NAME STRING TO SCAN FOR
0452 1153 : R10 = FILE VERSION NUMBER IF EXPLICIT, OR 0 IF DEFAULT TO LATEST VERSION
0452 1154 :
2B 11 0452 1155 BRB 120$ ;BEGIN LOOP AT BOTTOM
0454 1156 :
0454 1157 : R5 = ADDRESS OF NEXT RECORD
0454 1158 :
54 05 A5 9A 0454 1159 100$: MOVZBL DIR$B_NAMECOUNT(R5),R4 ;GET SIZE OF 'NAME.TYP' STRING
50 58 D0 0458 1166 MOVL R8,R0 ;DETERMINE SMALLER SIZE STRING
54 58 D1 045B 1167 CMPL R8,R4 ;ARE STRINGS SAME SIZE?
03 19 045E 1168 BLSS 105$ ;BR IF GOT THE SMALLER SIZE

```



```

06 A5 50 54 D0 0460 1169 MOVL R4,R0 ;GET SMALLER SIZE
69 50 29 0463 1170 105$: CMPC3 R0,(R9),DIR$T_NAME(R5) ;SEE IF STRINGS MATCH
19 19 0468 1171 BLSS 140$ ;BRANCH IF BEYOND WHERE NAME WOULD GO
07 12 046A 1172 BNEQ 106$ ;BRANCH TO KEEP SEARCHING
54 58 D1 046C 1173 CMPL R8,R4 ;CHECK THAT SIZES ARE SAME
1D 13 046F 1174 BEQL 200$ ;BRANCH IF STRINGS MATCH
10 19 0471 1175 BLSS 140$ ;BRANCH IF BEYOND WHERE NAME WOULD GO
50 65 3C 0473 1177 106$: MOVZWL DIR$W_SIZE(R5),R0 ;USING THE SIZE OF THIS RECORD
55 02 A540 9E 0476 1178 MOVAB 2(R5)(R0),R5 ;FORM ADDRESS OF NEXT RECORD
65 B5 047B 1179 110$: TSTW DIR$W_SIZE(R5) ;END OF BLOCK? (MARKED WITH -1)
D5 14 047D 1180 BGTR 100$ ;BRANCH IF NOT
06 56 57 F3 047F 1181 120$: AOBLEQ R7,R6,160$
50 0910 8F 3C 0483 1182 140$: MOVZWL #SS$ _NOSUCHFILE,R0 ;CANNOT FIND FILE
04 0488 1183 150$: RET
006D 30 0489 1184 160$: BSBW READ_DIR_LBN ;READ THE NEXT DIRECTORY LBN
ED 11 048C 1185 BRB 110$
048E 1186
048E 1187 : FOUND A MATCH OF FILE NAME AND TYPE
048E 1188
54 54 D6 048E 1189 200$: INCL R4 ;ROUND UP NAME COUNT
06 01 CA 0490 1190 BICL #1,R4 ;TO EVEN NUMBER OF BYTES
53 06 A544 9E 0493 1191 MOVAB DIR$T_NAME(R5)(R4),R3 ;ADDRESS OF FIRST VERSION ENTRY
50 65 3C 0498 1192 MOVZWL DIR$W_SIZE(R5),R0 ;SIZE OF THIS RECORD
55 02 A540 9E 049B 1193 MOVAB 2(R5)(R0),R5 ;FORM ADDRESS OF BEGINNING OF NEX RECORD
04A0 1194 ;WHICH IS ALSO THE END OF THE VERSIONS
5A D5 04A0 1195 TSTL R10 ;LATEST VERSION DESIRED?
11 13 04A2 1196 BEQL 240$ ;BRANCH IF YES, R3 IS ADDRESS OF
04A4 1197 ;DESIRED VERSION AND FILE ID
63 5A B1 04A4 1198 230$: CMPW R10,DIR$W_VERSION(R3) ;IS THIS THE RIGHT VERSION?
0C 13 04A7 1199 BEQL 240$ ;BRANCH IF YES
D8 1A 04A9 1200 BGTRU 140$ ;BRANCH IF PAST WHERE IT WOULD BE
53 08 C0 04AB 1201 ADDL #DIR$C_VERSION,R3 ;NEXT VERSION ENTRY
55 53 D1 04AE 1202 CMPL R3,R5 ;END OF RECORD?
F1 1F 04B1 1203 BLSSU 230$ ;BRANCH IF NOT, CHECK NEXT VERSION
C6 11 04B3 1204 BRB 110$ ;VERSION NOT IN THIS VERSION CHAIN
04B5 1205 ;SEE IF IT IS IN THE NEXT RECORD
04B5 1206
04B5 1207 : FOUND THE FILE ID, RETURN IT TO CALLER
04B5 1208
56 02 A3 7D 04B5 1209 240$: MOVQ DIR$W_FID(R3),R6 ;GET THE FILE ID
57 57 3C 04B9 1210 MOVZWL R7,R7
18 9C 02 A3 06 28 04BC 1211 MOVCL #6,DIR$W_FID(R3),@FILID(AP) ;AND RETURN IT TO THE CALLER
04C2 1212
04C2 1213 : SEE IF WE SHOULD MAKE A CACHE ENTRY FOR THIS LOOKUP
04C2 1214 : R6,R7 = FID
04C2 1215
04C2 1216 EXIT_FILID_FND:
5B D5 04C2 1217 TSTL R11 ;IS THE CACHE ENABLED?
2C 13 04C4 1218 BEQL 100$ ;BRANCH IF NOT, JUST RETURN THE FID
D6 AD 95 04C6 1219 TSTB FIL$T_DIR_NAM+ENTRY(FP) ;WAS LOOKUP FOR A DIRECTORY?
07 12 04C9 1220 BNEQ 20$ ;BRANCH IF YES, MAKE A CACHE ENTRY
F4 AD D5 04CB 1221 TSTL ENTRY_ADR(FP) ;CACHE HIT FOR THIS DIR HDR?
22 12 04CE 1222 BNEQ 100$ ;BRANCH IF YES
08 11 04D0 1223 BRB 30$ ;MAKE THE CACHE ENTRY FOR THIS DIR HDR
EE AD 56 D0 04D2 1224 20$: MOVL R6,FIL$A_DIR_OFID+ENTRY(FP) ;STORE THE FID FOUND
F2 AD 57 B0 04D6 1225 MOVW R7,FIL$A_DIR_OFID+4+ENTRY(FP)
58 08 AB D0 04DA 1226 30$: MOVL FIL$L_DIRNXT(R11),R8 ;GET OFFSET TO FREE SPACE

```

```

50 58 24 C1 04DE 1227 ADDL3 #FIL$C_DIR_SIZE,R8,R0 ;FORM OFFSET TO END OF NEW ENTRY
OC AB 50 D1 04E2 1228 CMPL R0,FIL$C_DIRMAX(R11) ;ENOUGH SPACE FOR NEW ENTRY?
OA 14 04E6 1229 BGTR 90$ ;BRANCH IF NOT
6B48 08 AB 50 D0 04E8 1230 MOVL R0,FIL$C_DIRNXT(R11) ;YES, ALLOCATE THE NEW ENTRY
DO AD 24 28 04EC 1231 MOVC3 #FIL$C_DIR_SIZE,ENTRY(FP),(R11)[R8] ;AND WRITE IT
50 01 3C 04F2 1232 90$:
04 04F5 1233 100$: MOVZWL #SS$ _NORMAL,R0 ;INDICATE SUCCESSFUL COMPLETION
04F6 1234 RET
04F6 1235 :
04F6 1236 : BAD DIRECTORY FILE
04F6 1237 :
FF08 31 04F6 1238 BADDIR1:
04F6 1239 BRW BADDIR

```



```

04F9 1241 .SBTTL READ_DIR_LBN - READ NEXT DIRECTORY LBN
04F9 1242 :++
04F9 1243 : FUNCTIONAL DESCRIPTION:
04F9 1244 :
04F9 1245 :     READ THE NEXT DIRECTORY LBN FROM THE DISK OR POINT AT
04F9 1246 :     THE CACHED COPY IF ONE IS PRESENT
04F9 1247 :
04F9 1248 : CALLING SEQUENCE:
04F9 1249 :
04F9 1250 :     BSBW    READ_DIR_LBN
04F9 1251 :
04F9 1252 : INPUT:
04F9 1253 :
04F9 1254 :     R6 = DESIRED LBN
04F9 1255 :     DIRBUF(AP) = BUFFER ADDRESS TO READ IT INTO
04F9 1256 :     CHAN(AP) = CHANNEL FOR FIL$RDWRTLBN
04F9 1257 :     DIR_BFCNT(FP) = COUNT OF BUFFERS REMAINING IN DIR LBN CACHE
04F9 1258 :     DIR_BUF(FP) = ADDRESS OF NEXT BUFFER IN DIR LBN CACHE
04F9 1259 :
04F9 1260 : OUTPUTS:
04F9 1261 :
04F9 1262 :     R5 = ADDRESS OF DESIRED DIRECTORY LBN
04F9 1263 :     RSB IF SUCCESSFUL
04F9 1264 :     RET WITH STATUS IN R0 IF ERROR
04F9 1265 :     R0, R1 DESTROYED, OTHERS PRESERVED
04F9 1266 :
04F9 1267 : --
04F9 1268 :
04F9 1269 READ_DIR_LBN:
04F9 1270     TSTL    DIR_BFCNT(FP)                ;ANYTHING LEFT IN DIR LBN CACHE?
04FC 1271     BEQL    20$                        ;BRANCH IF NOT
04FE 1272     DECL    DIR_BFCNT(FP)                ;COUNT ANOTHER BUFFER USED
0501 1273     MOVL    DIR_BUF(FP),R5                ;LOAD ADDRESS OF BUFFER
0505 1274     MOVAL   512(R5),DIR_BUF(FP)          ;AND POINT TO NEXT BUFFER IF ANY
050B 1275     10$:    RSB
050C 1276     :
050C 1277     : DIR LBN CACHE RAN OUT OF BLOCKS OR NEVER HAD ANY AT ALL
050C 1278     :
050C 1279     20$:    MOVL    DIRBUF(AP),R5                ;ADDRESS OF BUFFER TO READ INTO
0510 1280     READLBN CHAN(AP),R6,(R5)                ;READ THE DESIRED LBN
0510     ROTL    #9,#1,-(SP)
0514     MOVZWL  #10$_READLBLK,-(SP)
0517     PUSHAL  (R5)
0519     PUSHL   R6
051B     PUSHL  CHAN(AP)
051E     CALLS   #5,W^FIL$RDWRTLBN
0523 1281     BLBS   R0,10$                        ;BRANCH IF READ SUCCESSFULLY
0526 1282     RET                                ;RETURN ERROR STATUS

```

```

0527 1284 .SBTTL RDCHKFILHDR - READ AND CHECK FILE HEADER
0527 1285 :++
0527 1286 : FUNCTIONAL DESCRIPTION:
0527 1287 :
0527 1288 : RDCHKFILHDR READS AND VALIDATES A FILE HEADER GIVEN ITS FILE ID
0527 1289 : AND THE INDEX FILE HEADER AS RETURNED BY FIL$MOUNT.
0527 1290 :
0527 1291 : CALLING SEQUENCE:
0527 1292 :
0527 1293 : CALLG  ARGLIST,FIL$RDCHKFILHDR
0527 1294 :
0527 1295 : INPUT PARAMETERS:
0527 1296 :
0527 1297 :     CHAN(AP)          CHANNEL ON WHICH DEVICE IS ASSIGNED
0527 1298 :     UNUSED           UNUSED PARAMETER
0527 1299 :     IXFHDR(AP)       ADR OF INDEX FILE HEADER AS RETURNED BY FIL$MOUNT
0527 1300 :     FILHDR(AP)       ADDRESS OF 512 BYTE BUFFER FOR FILE HEADER
0527 1301 :     STATBLK(AP)      ADR OF 2 LONG WORD AREA TO RETURN STATISTICS BLOCK
0527 1302 :     FILID(AP)        ADDRESS OF 3 WORD FILE ID OF DESIRED FILE HEADER
0527 1303 :     RTRVPTLEN(AP) =  ADDRESS TO RETURN THE NUMBER OF
0527 1304 :                       BYTES OF RETRIEVAL POINTERS STORED
0527 1305 :
0527 1306 :     RTRVPTRBUF(AP) = ***** OPTIONAL PARAMETER *****
0527 1307 :                       ADDRESS OF RETRIEVAL POINTER
0527 1308 :                       BUFFER DESCRIPTOR. THIS PARAMETER
0527 1309 :                       IS PRESENT IF AND ONLY IF
0527 1310 :                       RTRVPTLEN IS PRESENT.
0527 1311 :                       THE RETRIEVAL POINTERS ARE RETURNED IN
0527 1312 :                       THE FORM 32 BIT BLOCK COUNT, 32 BIT LBN
0527 1313 :                       A ZERO BUFFER DESCRIPTOR ADDRESS OR A
0527 1314 :                       ZERO BUFFER ADDRESS MEANS DON'T
0527 1315 :                       RETURN RETRIEVAL POINTER INFO
0527 1316 :
0527 1317 : IMPLICIT INPUTS:
0527 1318 :
0527 1319 :     NONE
0527 1320 :
0527 1321 : OUTPUT PARAMETERS:
0527 1322 :
0527 1323 :     RO = SYSTEM STATUS CODE
0527 1324 :
0527 1325 : IMPLICIT OUTPUTS:
0527 1326 :
0527 1327 :     NONE
0527 1328 :
0527 1329 : COMPLETION CODES:
0527 1330 :
0527 1331 :     $$$_NORMAL      SUCCESSFUL COMPLETION
0527 1332 :
0527 1333 : SIDE EFFECTS:
0527 1334 :
0527 1335 :     NONE
0527 1336 :
0527 1337 : EQUATED SYMBOLS
0527 1338 :
0527 1339 :     OFFSETS FROM AP
0527 1340 :

```



```

00000000 0527 1341 ARGCNT = 0
00000004 0527 1342 CHAN = 4
0000000C 0527 1343 IXFHDR = 12
00000010 0527 1344 FILHDR = 16
00000014 0527 1345 STATBLK = 20
00000018 0527 1346 FILID = 24
0000001C 0527 1347 RTRVPTRLEN = 28 ; OPTIONAL PARAMETER
00000020 0527 1348 RTRVPTRBUF = 32 ; PRESENT IF AND ONLY IF RTRVPTRLEN IS

0527 1349 :
0527 1350 : OFFSETS FROM FP
0527 1351 :
0527 1352 : $OFFSET 0,NEGATIVE,<-
0527 1353 : <HDCNT>,- ;COUNT OF FILE HEADERS READ
0527 1354 : <TMPRTRVLEN>,- ;TEMP RTRV PTR BYTE COUNT
0527 1355 : <TMPRTRVDSC,8>- ;TEMP RTRV PTR BUFFER DESCRIPTOR
0527 1356 :
FFFC HDRCNT:
FFF8 TMPRTRVLEN:
FFF0 TMPRTRVDSC:
0527 1357 :
0527 1358 :--
0527 1359 :
0527 1360 FIL$RDCHKFILHDR::
0527 1361 .WORD ^M<R2,R3,R4,R5,R6>
7E 01 CE 0529 1362 MNEGL #1,-(SP) ;COUNT OF HEADER BLOCKS READ
7E 7E D4 052C 1363 CLRL -(SP) ;INIT RTRV PTR BYTE COUNT
7E 7E 7C 052E 1364 CLRQ -(SP) ;ASSUME NO RTRV PTR BUFFER
08 6C D1 0530 1365 CMPL ARGCNT(AP),#RTRVPTRBUF/4 ;RTRV PTR PARAMS PRESENT?
OD 19 0533 1366 BLSS 2$ ;BRANCH IF NOT
50 20 AC D0 0535 1367 MOVL RTRVPTRBUF(AP),R0 ;RTRV BUFFER DESCRIPTOR ADDRESS
07 13 0539 1368 BEQL 2$ ;BRANCH IF NONE SPECIFIED
FO AD 60 7D 053B 1369 MOVQ (R0),TMPRTRVDSC(FP) ;MAKE COPY OF BUF DESCRIPTOR
1C BC D4 053F 1370 CLRL @RTRVPTRLEN(AP) ;INIT RETURNED BYTE COUNT
0542 1371
0542 1372
52 0C AC 7D 0542 1373 2$: ASSUME FILHDR EQ IXFHDR+4
MOVQ IXFHDR(AP),R2 ;R2 = INDEX FILE HEADER ADDRESS
0546 1374 ;R3 = FILE HEADER ADDRESS
0546 1375 ASSUME FILID EQ STATBLK+4
54 14 AC 7D 0546 1376 MOVQ STATBLK(AP),R4 ;R4 = RETURN STATBLK ADDRESS
054A 1377 ;R5 = ADDRESS OF FILE ID
7E 65 7D 054A 1378 MOVQ (R5),-(SP) ;COPY FILE ID TO WRITABLE SCRATCH
55 5E D0 054D 1379 MOVL SP,R5 ;AND REMEMBER ITS ADDRESS
56 7E 7E 0550 1380 MOVAQ -(SP),R6 ;RESERVE SCRATCH STATISTICS BLOCK
0553 1381 ;AND SAVE ITS ADDRESS IN R6
64 7C 0553 1382 CLRQ (R4) ;INIT THE RETURN STAT BLOCK
64 D7 0555 1383 DECL (R4) ;-1 LBN MEANS NOT YET STORED
7E 65 3C 0557 1384 5$: MOVZWL (R5),-(SP) ;FILE NUMBER TO LONG WORD
05 06 A2 0A E1 055A 1385 BBC #FH2$V BIGFILNUM,FH2$W-STRUCLEV(R2),10$ ;BRANCH IF NO BIG FIL NUM
02 AE 05 A5 90 055F 1386 MOVQ FID$B_NMX(R5),2(SP) ;HIGH 8 BITS OF RVN COMPLETE FILE NUMBER
50 8E D0 0564 1387 10$: MOVL (SP)+,R0 ;IF FILE ID IS ZERO,
03 12 0567 1388 BNEQ 12$
006D 31 0569 1389 BRW 40$ ;THEN READ LAST HEADER BLOCK
51 01FE C2 3C 056C 1390 12$: MOVZWL FH2$W_VBNOFFSET(R2),R1 ;RECOVER VBN OFFSET FROM INDEX FILE HEADER
50 51 C0 0571 1391 ADDL R1,R0 ;ADD VBN OFFSET TO FORM INDEX FILE VBN
FC AD D6 0574 1392 INCL HDRCNT(FP) ;COUNT EACH HEADER READ
62 DF 0577 1393 READVBN CHAN(AP),R0,(R3),(R2) ;READ THE FILE HEADER
PUSHAL (R2)

```

	63	DF	0579		PUSHAL	(R3)	
	50	DD	057B		PUSHL	R0	
	04	AC	057D		PUSHL	CHAN(AP)	
05F2'CF	04	FB	0580		CALLS	#4,W*FIL\$READVBN	
SD FC AD	62	E9	0585	1394	R0,50\$		:BRANCH IF ERROR
50	1F	E0	0588	1395	#31,HDRCNT(FP),50\$		:BRANCH IF JUST RE-READING MAIN HEADER
51	55	D0	058D	1396	R5,R0		:GET FILE ID ADDRESS
	53	D0	0590	1397	R3,R1		:ADDRESS OF FILE HEADER
	0153	30	0593	1398	FIL\$CHKFILHDR		:CHECK THE FILE HEADER
52	OC AC	D0	0596	1399	IXFHDR(AP),R2		:INDEX FILE HEADER ADDRESS
	FO AD	DF	059A	1400	TMPTRVDSC(FP)		:RTRV PTR BUF DESCRIPTOR
	F8 AD	DF	059D	1401	TMPTRVLEN(FP)		:ADDRESS TO RETURN BYTE COUNT
	56	DD	05A0	1402	R6		:ADDRESS OF SCRATCH STAT BLOCK
	53	DD	05A2	1403	R3		:ADDRESS OF FILE HEADER
0685'CF	04	FB	05A4	1404	CALLS	#4,W*FIL\$STATBLK	:READ STATISTICS BLOCK
51	F8 AD	D0	05A9	1405	TMPTRVLEN(FP),R1		:ANY RTRV PTR INFO TO RETURN?
	16	13	05AD	1406	BEQL	16\$	:ZERO IF NONE REQUESTED
1C BC	51	C0	05AF	1407	ADDL	R1,@RTRVPTRLEN(AP)	:ACCUMULATE RTRVPTR BYTE COUNT
FO AD	51	D1	05B3	1408	CMLP	R1,TMPTRVDSC(FP)	:MORE SPACE NEEDED THAN WOULD FIT?
	04	15	05B7	1409	BLEQ	14\$	:BRANCH IF NOT
51	FO AD	D0	05B9	1410	MOVL	TMPTRVDSC(FP),R1	:SAY WE USED IT ALL UP
F4 AD	51	C0	05BD	1411	ADDL	R1,TMPTRVDSC+4(FP)	:GET NEW STARTING ADDRESS
FO AD	51	C2	05C1	1412	SUBL	R1,TMPTRVDSC(FP)	:AND CALC NEW SIZE REMAINING
	51	D2	05C5	1413	MCOML	(R4),R1	:SEE IF START LBN HAS BEEN SET
	03	12	05C8	1414	BNEQ	20\$	:BRANCH IF IT HAS
04	64	D0	05CA	1415	MOVL	(R6),(R4)	:SET IT ONCE ONLY
A4	04 A6	C0	05CD	1416	ADDL	4(R6),4(R4)	:ADD IN THE SIZE FROM THIS HEADER
65	OE A3	7D	05D2	1417	MOVQ	FH2\$W_EXT_FID(R3),(R5)	:GET EXTENSION FILE ID IF ANY
	FF7E	31	05D6	1418	BRW	5\$	:READ THIS HEADER IF ANY
			05D9	1419			
			05D9	1420			: LAST FILE HEADER READ, SEE IF MUST RE-READ THE ORIGINAL HEADER
			05D9	1421			
	FC AD	D5	05D9	1422	TSTL	HDRCNT(FP)	:WAS -1, BUMPED ONCE PER READVBN
	09	15	05DC	1423	BLEQ	45\$	:BRANCH IF STILL HAVE MASTER FILE HEADER
65	18 BC	7D	05DE	1424	MOVQ	@FILID(AP),(R5)	:ORIGINAL FILE ID AGAIN
EF FC AD	1F	E3	05E2	1425	BBCS	#31,HDRCNT(FP),30\$	:SET SIGN BIT AND GO READ ORIGINAL HEADER
50	01	3C	05E7	1426	MOVZWL	#SS\$_NORMAL,R0	:RETURN SUCCESS STATUS
		04	05EA	1427	RET		



```

05EB 1429 .SBTTL READVBN, WRITEVBN - READ/WRITE VIRTUAL BLOCK
05EB 1430 :++
05EB 1431 : FUNCTIONAL DESCRIPTION:
05EB 1432 :
05EB 1433 :     THESE ROUTINES READ OR WRITE A VIRTUAL BLOCK FROM A FILE.
05EB 1434 :     VOLUME IS SPECIFIED BY THE CHANNEL TO WHICH IT IS ASSIGNED, AND THE
05EB 1435 :     FILE IS SPECIFIED BY THE ADDRESS OF ITS FILE HEADER WHICH WAS PREVIOUSLY
05EB 1436 :     READ BY A CALL TO FIL$RDFILHDR.
05EB 1437 :
05EB 1438 : CALLING SEQUENCE:
05EB 1439 :
05EB 1440 :     CALLG  ARGLIST,FIL$READVBN
05EB 1441 :     CALLG  ARGLIST,FIL$WRITEVBN
05EB 1442 :
05EB 1443 : INPUT PARAMETERS:
05EB 1444 :
05EB 1445 :     CHAN(AP)      =                ;CHANNEL TO WHICH VOLUME IS ASSIGNED
05EB 1446 :     VBN(AP)       =                ;DESIRED VIRTUAL BLOCK NUMBER
05EB 1447 :     BUFADR(AP)    =                ;ADDRESS OF BUFFER TO READ INTO
05EB 1448 :     FILHDR(AP)    =                ;ADDRESS OF FILE HEADER
05EB 1449 :
05EB 1450 : IMPLICIT INPUTS:
05EB 1451 :
05EB 1452 :     NONE
05EB 1453 :
05EB 1454 : OUTPUT PARAMETERS:
05EB 1455 :
05EB 1456 :     R0 = SYSTEM STATUS CODE
05EB 1457 :
05EB 1458 : IMPLICIT OUTPUTS:
05EB 1459 :
05EB 1460 :     NONE
05EB 1461 :
05EB 1462 : COMPLETION CODES:
05EB 1463 :
05EB 1464 :     $$$_NORMAL                SUCCESSFUL RETURN
05EB 1465 :     $$$_ENDOFFILE             SPECIFIED VBN BEYOND END OF FILE
05EB 1466 :
05EB 1467 : SIDE EFFECTS:
05EB 1468 :
05EB 1469 :     NONE
05EB 1470 :
05EB 1471 : EQUATED SYMBOLS:
05EB 1472 :
05EB 1473 :     OFFSET FROM AP
05EB 1474 :
05EB 1475 :
05EB 1476 :     CHAN            =            4      ;CHANNEL TO WHICH VOLUME IS ASSIGNED
05EB 1477 :     VBN             =            8      ;VIRTUAL BLOCK NUMBER
05EB 1478 :     BUFADR          =           12      ;BUFFER ADDRESS TO READ INTO
05EB 1479 :     FILHDR          =           16      ;ADDRESS OF FILE HEADER
05EB 1480 :
05EB 1481 :     OFFSETS FROM FP
05EB 1482 :
05EB 1483 :     IOFUNCTION      =           -4      ;SAVED I/O FUNCTION CODE
05EB 1484 :
05EB 1485 : --

```

00000004  
00000008  
0000000C  
00000010

FFFFFFFFC

```

05EB 1486
05EB 1487 FIL$WRITEVBN::
7E 20 003C 05EB 1488 .WORD ^M<R2,R3,R4,R5>
05 11 05ED 1489 MOVZWL #IOS$ WRITEBLK,-(SP)
05F0 1490 BRB RDWRTVBN
05F2 1491
05F2 1492 FIL$READVBN::
7E 21 003C 05F2 1493 .WORD ^M<R2,R3,R4,R5>
05F4 1494 MOVZWL #IOS$ READBLK,-(SP)
05F7 1495
05F7 1496 RDWRTVBN:
55 10 AC D0 05F7 1497 MOVL FILHDR(AP),R5 ;BASE ADR OF FILE HEADER
31 10 05FB 1498 BSBB INIRTRVPTRSCAN ;SET UP TO SCAN RETRIEVAL POINTERS
05FD 1499
05FD 1500 : R4 = POINTER TO FIRST RETRIEVAL POINTER,
05FD 1501 : R5 = POINTER TO FIRST BYTE BEYOND LAST RETRIEVAL POINTER
05FD 1502 : LOOP THROUGH RETRIEVAL POINTERS TO FIND THE ONE WHICH CONTAINS THE DESIRED VBN
05FD 1503
53 08 AC 01 C3 05FD 1504 SUBL3 #1,VBN(AP),R3 ;VBN BASE 0 TO LOOK FOR
3B 10 0602 1505 20$: BSBB GETRTRVPTR ;FETCH NEXT RETRIEVAL POINTER
50 53 D1 0604 1506 CMPL R3,R0 ;IS VBN IN THIS RETRIEVAL POINTER
OE 19 0607 1507 BLSS 40$ ;BRANCH IF YES
53 50 C2 0609 1508 SUBL R0,R3 ;PASS OVER THAT MANY VBN'S
55 54 D1 060C 1509 CMPL R4,R5 ;ANY MORE RETRIEVAL POINTERS?
F1 1F 060F 1510 BLSSU 20$ ;BRANCH IF YES
50 0870 8F 3C 0611 1511 MOVZWL #SS$_ENDOFFILE,R0 ;RETURN END OF FILE INDICATION
04 0616 1512 RET
0617 1513
0617 1514 : VBN IS IN THIS RETRIEVAL POINTER, R1 = STARTING LBN
0617 1515
7E 01 09 9C 0617 1516 40$: ROTL #9,#1,-(SP) ;NUMBER OF BYTES TO READ/WRITE
FC AD DD 061B 1517 PUSHL IOFUNCTION(FP) ;FUNCTION CODE
OC AC DD 061E 1518 PUSHL BUFADR(AP) ;BUFFER TO TRANSFER TO/FROM
7E 51 53 C1 0621 1519 ADDL3 R3,R1,-(SP) ;LBN
04 AC DD 0625 1520 PUSHL CHAN(AP) ;CHANNEL
0000'CF 05 FB 0628 1521 CALLS #5,W^FIL$RDWRTLBN ;TRANSFER THE BLOCK
04 062D 1522 RET

```



```

062E 1524 .SBTTL INIRTRVPTRSCAN - INITIALIZE RETRIEVAL POINTER SCAN
062E 1525 :++
062E 1526 : FUNCTIONAL DESCRIPTION:
062E 1527 :
062E 1528 : LOCATE START AND END OF RETRIEVAL POINTERS IN A FILE HEADER.
062E 1529 :
062E 1530 : CALLING SEQUENCE:
062E 1531 :
062E 1532 : BSBW INIRTRVPTRSCAN
062E 1533 :
062E 1534 : INPUT:
062E 1535 :
062E 1536 : R5 = FILE HEADER ADDRESS
062E 1537 :
062E 1538 : OUTPUT:
062E 1539 :
062E 1540 : R4 = ADDRESS OF 1ST RETRIEVAL POINTER
062E 1541 : R5 = ADDRESS OF FIRST BYTE BEYOND LAST RETREIVAL POINTER
062E 1542 :
062E 1543 :--
062E 1544 :
062E 1545 INIRTRVPTRSCAN:
50 01 A5 9A 062E 1546 MOVZBL FH2$B MPOFFSET(R5),R0 ;WORD OFFSET TO MAP AREA
54 6540 3E 0632 1547 MOVAW (R5)[R0],R4 ;BASE ADR OF MAP AREA
55 3A A5 9A 0636 1548 MOVZBL FH2$B MAP INUSE(R5),R5 ;NO. OF WORDS OF RTRV PTRS IN USE
55 6445 3E 063A 1549 10$: MOVAW (R4)[R5],R5 ;ADR JUST BEYOND LAST VALID RTRV PTR
05 063E 1550 RSB

```



```

063F 1552 .SBTTL GETRTRVPTR - CONVERT NEXT RETRIEVAL POINTER
063F 1553 :++
063F 1554 : FUNCTIONAL DESCRIPTION:
063F 1555 :
063F 1556 : CONVERT NEXT RETRIEVAL POINTER TO NUMBER OF BLOCKS COVERED BY
063F 1557 : POINTER AND STARTING LBN.
063F 1558 :
063F 1559 : CALLING SEQUENCE:
063F 1560 :
063F 1561 : BSBW GETRTRVPTR
063F 1562 :
063F 1563 : INPUTS:
063F 1564 :
063F 1565 : R4 = ADDRESS OF NEXT RETRIEVAL POINTER
063F 1566 :
063F 1567 : OUTPUTS:
063F 1568 :
063F 1569 : R0 = NUMBER OF BLOCKS COVERED BY THE RETRIEVAL POINTER
063F 1570 : R1 = STARTING LOGICAL BLOCK NUMBER
063F 1571 : R2,R3 PRESERVED
063F 1572 :
063F 1573 :--
063F 1574 :
063F 1575 GETRTRVPTR:
063F 1576 :
063F 1577 : STRUCTURE LEVEL 2 RETRIEVAL POINTERS
063F 1578 : BITS 14:15 = RETRIEVAL POINTER FORMAT
063F 1579 :
50 64 02 0E EF 063F 1580 20$: EXTZV #FM2$V_FORMAT,#FM2$S_FORMAT,(R4),R0 ;FORMAT TO R0
0644 1581 CASE RO,<-
0644 1582 PLACEMENT,- ;PLACEMENT FORMAT
0644 1583 FORMAT1,- ;FORMAT 1
0644 1584 FORMAT2- ;FORMAT 2
0644 1585 >
064E 1586 :
064E 1587 : FORMAT 3 = 8 BYTES
064E 1588 :
064E 1589 : BITS 0:13 = BITS 16:29 OF COUNT - 1
064E 1590 : BITS 14:15 = FORMAT = 3
064E 1591 : BYTES 2-3 = BITS 0:15 OF COUNT - 1
064E 1592 : BYTES 4-7 = LOGICAL BLOCK NUMBER
064E 1593 :
064E 1594 FORMAT3:
064E 1595 ROTL #16,(R4)+,R0 ;FORM COUNT - 1
50 02 84 10 9C 0652 1596 INSV #0,#30,#2,R0 ;ZERO HIGH 2 BITS
0657 1597 MOVL (R4)+,R1 ;GET LBN
065A 1598 BRB INCRSB ;INCREMENT COUNT AND EXIT
065C 1599 :
065C 1600 : PLACEMENT CONTROL - THIS IS NOT A RETRIEVAL POINTER, RATHER IT
065C 1601 : CONSISTS OF 2 BYTES OF PLACEMENT INFORMATION. TREAT AS IF 0
065C 1602 : LENGTH RETRIEVAL POINTER.
065C 1603 : R0 = 0
065C 1604 :
065C 1605 PLACEMENT:
51 01 CE 065C 1606 MNEGL #1,R1 ;IMPOSSIBLE LBN
54 02 CO 065F 1607 ADDL #2,R4 ;BUMP THE POINTER
0662 1608 CLRL R0 ;CLEAR BLOCK COUNT

```



```

05 0664 1609      RSB
    0665 1610      :
    0665 1611      : FORMAT 1 = 4 BYTES
    0665 1612      : BITS 0:7 = COUNT - 1
    0665 1613      : BITS 8:13 = BITS 16:21 OF LOGICAL BLOCK NUMBER
    0665 1614      : BYTES 2-3 = BITS 0:15 OF LOGICAL BLOCK NUMBER
    0665 1615      :
    0665 1616      : FORMAT1:
51 50 50 84 D0 0665 1617      MOVL      (R4)+,R0      ;FETCH ENTIRE RETRIEVAL POINTER
    50 06 08 EF 0668 1618      EXTZV     #FM2$V_HIGHLBN,#FM2$S_HIGHLBN,R0,R1 ;FETCH HIGH LBN BITS
    50 50 10 79 066D 1619      ASHQ      #16,R0,R0      ;FORM R1 = LBN
    50 FC A4 9A 0671 1620      MOVZBL    -4(R4),R0      ;REFETCH COUNT - 1
    50 50 D6 0675 1621      INCRSB:
    05 0677 1622      INCL      R0      ;FORM COUNT
    0678 1623      RSB      ;AND RETURN
    0678 1624      :
    0678 1625      : FORMAT 2 = 6 BYTES
    0678 1626      :
    0678 1627      : BITS 0:13 = COUNT - 1
    0678 1628      : BITS 14:15 = FORMAT = 2
    0678 1629      : BYTES 2-5 = LBN
    0678 1630      :
    0678 1631      : FORMAT2:
50 50 50 84 3C 0678 1632      MOVZWL    (R4)+,R0      ;FETCH COUNT - 1 AND FORMAT BITS
    50 0E 00 EF 067B 1633      EXTZV     #FM2$V_COUNT2,#FM2$S_COUNT2,R0,R0 ;COUNT - 1
    51 84 D0 0680 1634      MOVL      (R4)+,R1      ;LBN
    F0 11 0683 1635      BRB      INCRSB      ;INCREMENT COUNT AND RETURN

```



```

0685 1637 .SBTTL STATBLK - GET FILE STATISTICS BLOCK
0685 1638 :++
0685 1639 : FUNCTIONAL DESCRIPTION:
0685 1640 :
0685 1641 :     GIVEN A FILE HEADER, RETURN THE FILE STATISTICS BLOCK
0685 1642 :     AND OPTIONALLY RETURN THE RETRIEVAL POINTERS
0685 1643 :
0685 1644 : CALLING SEQUENCE:
0685 1645 :
0685 1646 :     CALLG  ARGLIST,FIL$STATBLK
0685 1647 :
0685 1648 : INPUT PARAMETERS:
0685 1649 :
0685 1650 :     FILHDR(AP)      =           ;ADDRESS OF THE FILE HEADER
0685 1651 :     STATBLK(AP)     =           ;ADDRESS TO RETURN STATISTICS BLOCK
0685 1652 :     RTRVPTRLEN(AP)  =           ;ADDRESS TO RETURN THE NUMBER OF
0685 1653 :                               ;BYTES OF RETRIEVAL POINTERS
0685 1654 :                               ;FOUND IN THE FILE HEADER(S).
0685 1655 :                               ;***** OPTIONAL PARAMETER *****
0685 1656 :     RTRVPTRBUF(AP)  =           ;ADDRESS OF RETRIEVAL POINTER
0685 1657 :                               ;BUFFER DESCRIPTOR. THIS PARAMETER
0685 1658 :                               ;IS PRESENT IF AND ONLY IF
0685 1659 :                               ;RTRVPTRLEN IS PRESENT.
0685 1660 :                               ;ZERO DESCRIPTOR ADDRESS OR ZERO
0685 1661 :                               ;BUFFER ADDRESS MEANS DON'T
0685 1662 :                               ;RETURN RETRIEVAL POINTER INFO
0685 1663 :
0685 1664 : IMPLICIT INPUTS:
0685 1665 :
0685 1666 :     NONE
0685 1667 :
0685 1668 : OUTPUT PARAMETERS:
0685 1669 :
0685 1670 :     R0 = SYSTEM STATUS CODE
0685 1671 :     STATBLK CONTAINS 2 LONGWORDS
0685 1672 :     LBN OF 1ST BLOCK IF CONTIGUOUS OR ZERO IF NOT
0685 1673 :     SIZE OF FILE IN BLOCKS
0685 1674 :     RTRVPTRLEN RECEIVES THE NUMBER OF BYTES OF RETRIEVAL POINTER
0685 1675 :     INFORMATION THAT WOULD HAVE BEEN STORED IN THE RETRIEVAL
0685 1676 :     POINTER BUFFER GIVEN A LARGE ENOUGH BUFFER.
0685 1677 :     THE RETRIEVAL POINTER BUFFER RECEIVES NORMALIZED RETRIEVAL
0685 1678 :     POINTERS IN THE FORMAT 32 BIT COUNT, 32 BIT STARTING LBN
0685 1679 :
0685 1680 : IMPLICIT OUTPUTS:
0685 1681 :
0685 1682 :     NONE
0685 1683 :
0685 1684 : COMPLETION CODES:
0685 1685 :
0685 1686 :     SS$_NORMAL          SUCCESSFUL COMPLETION
0685 1687 :
0685 1688 : SIDE EFFECTS:
0685 1689 :
0685 1690 :     NONE
0685 1691 :
0685 1692 :
0685 1693 : EQUATED SYMBOLS:

```



```

0685 1694 :
0685 1695 : OFFSETS FROM AP
0685 1696 :
00000000 0685 1697 ARGCNT = 0 ;NUMBER OF ARGUMENTS
00000004 0685 1698 FILHDR = 4 ;ADDRESS OF FILE HEADER
00000008 0685 1699 STATBLK = 8 ;ADDRESS TO RETURN STATISTICS BLOCK
0000000C 0685 1700 RTRVPTRLEN = 12 ;ADDRESS TO RETURN COUNT OF BYTES
00000010 0685 1701 RTRVPTRBUF = 16 ;STORED IN THE RETRIEVAL POINTER BUFFER
0685 1702 ;ADDRESS OF RETRIEVAL POINTER
0685 1703 ;BUFFER DESCRIPTOR
0685 1704 :
0685 1705 :--
0685 1706
0685 1707 FIL$STATBLK::
00FC 0685 1708 .WORD ^M<R2,R3,R4,R5,R6,R7>
04 56 7C 0687 1709 CLRQ R6 ;ASSUME NOT DOING RETRIEVAL POINTERS
06 6C D1 0689 1710 CMPL ARGCNT(AP),#RTRVPTRBUF/4 ;RTRV PTR PARAMS PRESENT?
50 10 AC D0 068C 1711 BLSS 5$ ;BRANCH IF NOT
56 60 7D 0692 1712 BEQL RTRVPTRBUF(AP),R0 ;ADDRESS OF BUFFER DESCRIPTOR
56 07 CA 0694 1713 BEQL 5$ ;BRANCH IF NOT SPECIFIED
OC BC D4 0697 1714 MOVQ (R0),R6 ;R6 = MAX SIZE, R7 = BUFFER ADR
55 04 AC D0 069A 1715 BICL #7,R6 ;EVEN MULTIPLE OF 8 BYTES
50 34 A5 9A 069D 1716 CLRL @RTRVPTRLEN(AP) ;INIT RETURN BYTE COUNT
7E 50 01 07 EF 06A1 1717 5$: MOVL FILHDR(AP),R5 ;ADDRESS OF FILE HEADER
FF81 30 06AA 1718 10$: MOVZBL FH2$ FILECHAR(R5),R0 ;FILE CHARACTERISTICS IF LEVEL 2
53 D4 06AD 1719 EXTZV #FH2$ CONTIG,#1,R0,-(SP) ;CONTIGUOUS BIT TO TOP OF STACK
29 11 06AF 1720 BSBW INIRTRVPTRSCAN ;INIT FOR SCAN OF RETRIEVAL POINTERS
FF8B 30 06B1 1721 CLRL R3 ;INIT REGISTER TO COUNT BLOCKS
53 D5 06B4 1722 BRB 50$ ;START AT BOTTOM OF LOOP IN CASE
OA 12 06B8 1723 20$: BSBW GETRTRVPTR ;FILE HAS NO RETRIEVAL POINTERS
06B8 1724 TSTL R3 ;GET THE NEXT RETRIEVAL POINTER
06B8 1725 BNEQ 40$ ;IS THIS FIRST RTRV PTR?
06B8 1726 ;BRANCH IF ALREADY COUNTED SOME
06B8 1727 : FIRST RETRIEVAL POINTER
06B8 1728 :
06B8 1729 :
50 D5 06B8 1730 TSTL R0 ;IGNORE EMPTY ONES
1E 13 06BA 1731 BEQL 50$
03 6E E9 06BC 1732 BLBC (SP),40$
06BF 1733
6E 51 D0 06BF 1734 MOVL R1,(SP) ;BRANCH IF FILE NOT CONTIGUOUS
53 50 C0 06C2 1735 40$: ADDL R0,R3 ;0(SP) = 0 IN THIS CASE
56 D5 06C5 1736 TSTL R6 ;SET LBN OF 1ST NON-ZERO RTRV PTR
09 13 06C7 1737 BEQL 45$ ;ACCUMULATE COUNT OF BLOCKS
87 50 D0 06C9 1738 MOVL R0,(R7)+ ;ANY MORE ROOM FOR RTRV PTRS?
87 51 D0 06CC 1739 MOVL R1,(R7)+ ;BRANCH IF NO MORE BUFFER SPACE
56 08 C2 06CF 1740 SUBL #8,R6 ;STORE SIZE OF RETRIEVAL POINTER
57 D5 06D2 1741 45$: TSTL R7 ;AND STORE LBN
04 13 06D4 1742 BEQL 50$ ;USED 8 MORE BYTES OF SPACE
OC BC 08 C0 06D6 1743 ADDL #8,@RTRVPTRLEN(AP) ;DOES CALLER WANT RTRV PTR INFO?
55 54 D1 06DA 1744 50$: CMPL R4,R5 ;BRANCH IF NOT, DON'T COUNT POINTERS
D2 1F 06DD 1745 BLSSU 20$ ;UPDATE RTRV PTR BYTE COUNT
04 BA 06DF 1746 POPR #^M<R2> ;ANY MORE RETRIEVAL POINTERS?
08 BC 52 7D 06E1 1747 MOVQ R2,@STATBLK(AP) ;BRANCH IF YES
50 01 3C 06E5 1748 MOVZWL #SS$_NORMAL,R0 ;GET SAVED STARTING LBN
04 06E8 1749 RET ;RETURN THE STATISTICS BLOCK
;SUCCESSFUL COMPLETION

```



```

06E9 1751 .SBTTL FIL$CHKFILHDR - CHECK FILE HEADER VALIDITY
06E9 1752 :++
06E9 1753 : FUNCTIONAL DESCRIPTION:
06E9 1754 :
06E9 1755 : CHECK THE VALIDITY OF A FILE HEADER
06E9 1756 :
06E9 1757 : CALLING SEQUENCE:
06E9 1758 :
06E9 1759 : BSBW FIL$CHKFILHDR
06E9 1760 :
06E9 1761 : INPUT PARAMETERS:
06E9 1762 :
06E9 1763 : R0 = ADDRESS OF FILE ID
06E9 1764 : R1 = ADDRESS OF FILE HEADER
06E9 1765 :
06E9 1766 : IMPLICIT INPUTS:
06E9 1767 :
06E9 1768 : NONE
06E9 1769 :
06E9 1770 : OUTPUT PARAMETERS:
06E9 1771 :
06E9 1772 : RSB TO CALLER IF FILE HEADER VALID
06E9 1773 : RET IF NOT VALID WITH R0 = ERROR STATUS
06E9 1774 :
06E9 1775 : IMPLICIT OUTPUTS:
06E9 1776 :
06E9 1777 : NONE
06E9 1778 :
06E9 1779 : COMPLETION CODES:
06E9 1780 :
06E9 1781 : SSS_BADFILEHDR FILE ID CODES DON'T MATCH
06E9 1782 : SSS_NOSUCHFILE FILE IS MARKED AS DELETED
06E9 1783 :
06E9 1784 : SIDE EFFECTS:
06E9 1785 :
06E9 1786 : NONE
06E9 1787 :
06E9 1788 : --
06E9 1789 :
06E9 1790 FIL$CHKFILHDR:
02 07 A1 91 06E9 1791 CMPB FH2$B_STRUCLEV(R1),#2 ;IS THIS STRUCTURE LEVEL 2?
1E 12 06ED 1792 BNEQ 30$ ;BR IF NOT, REPORT ERROR
06EF 1793 :
06EF 1794 : STRUCTURE LEVEL 2
06EF 1795 :
7E 0C A1 3C 06EF 1796 10$: MOVZWL FH2$W_FID_RVN(R1),-(SP) ;PUSH RELATIVE VOLUME NUMBER
7E 08 A1 D0 06F3 1797 MOVL FH2$W_FID_NUM(R1),-(SP) ;PUSH FILE ID ON STACK
6E 18 B5 06F7 1798 15$: TSTW (SP) ;FILE DELETED?
8E 80 D1 06F9 1799 BEQL 40$ ;BRANCH IF YES
0D 12 06FB 1800 CMPL (R0)+,(SP)+ ;FILE NUM AND FILE SEQ NUM AGREE?
6E 05 D5 0700 1801 BNEQ 30$ ;BRANCH IF NOT, BAD HEADER
05 17 0702 1802 TSTL (SP) ;CHECKING RVN?
6E 60 B1 0704 1803 BLSS 20$ ;BRANCH IF NOT
04 12 0707 1804 CMPW (R0),(SP) ;RELATIVE VOLUME NUMBER AND
01 BA 0709 1805 ;FILE NUMBER EXTENSION AGREE
1806 ;BRANCH IF NOT
1807 20$: POPR #*M<R0> ;CLEAN OFF STACK

```



FILEREADUV1  
V03-003

H 10  
- MICRO-VAX I FILES-11 LEVEL 2 FILE READ 10-AUG-1984 18:05:11 VAX/VMS Macro V04-00  
FIL\$CHKFILHDR - CHECK FILE HEADER VALIDI 9-JUL-1984 11:44:50 FILEREAD.MAR;1

Page 38  
(17)

50	0810	OC	11	070B	1808	BRB	FIL\$CHECKSUM	:GO VERIFY THE CHECKSUM
		8F	3C	070D	1809	MOVZWL	#SS\$_BADFILEHDR,R0	:THIS HEADER IS BAD
			04	0712	1810	RET		
50	0910	8F	3C	0713	1811	MOVZWL	#SS\$_NOSUCHFILE,R0	:DELETED FILE
			04	0718	1812	RET		

```

0719 1814 .SBTTL CHECKSUM - VALIDATE A CHECKSUM
0719 1815 :++
0719 1816 : FUNCTIONAL DESCRIPTION:
0719 1817 :
0719 1818 : THIS ROUTINE CALCULATES AND CHECKS THE FILE11 CHECKSUM FOR
0719 1819 : FILE HEADERS AND THE HOMEBLOCK.
0719 1820 :
0719 1821 : CALLING SEQUENCE:
0719 1822 :
0719 1823 : BSBW FIL$CHECKSUM ;CHECK FILE HEADER CHECKSUM
0719 1824 : BSBW FIL$CHECKSUM1 ;CHECK SPECIFIED NO. OF WORDS IN RO
0719 1825 :
0719 1826 : INPUT PARAMETERS:
0719 1827 :
0719 1828 : RO = NO. OF WORDS TO CHECK IF ENTERING AT CHECKSUM1
0719 1829 : R1 = ADDRESS OF BUFFER TO CHECK
0719 1830 :
0719 1831 : IMPLICIT INPUTS:
0719 1832 :
0719 1833 : NONE
0719 1834 :
0719 1835 : OUTPUT PARAMETERS:
0719 1836 :
0719 1837 : RSB TO CALLER IF CHECKSUM IS OK
0719 1838 : RET TO TOP LEVEL WITH ERROR CODE IN RO IF CHECKSUM IS WRONG
0719 1839 :
0719 1840 : IMPLICIT OUTPUTS:
0719 1841 :
0719 1842 : NONE
0719 1843 :
0719 1844 : COMPLETION CODES:
0719 1845 :
0719 1846 : NONE
0719 1847 :
0719 1848 : SIDE EFFECTS:
0719 1849 :
0719 1850 : NONE
0719 1851 :
0719 1852 : --
0719 1853 :
0719 1854 FIL$CHECKSUM:
50 00FF 8F 3C 0719 1855 MOVZWL #FH2$W_CHECKSUM@-1,RO ;NO. OF WORDS TO CHECK
0719 1856 FIL$CHECKSUM1:
0719 1857 CLRL R2 ;INIT THE SUM
0719 1858 10$:
0719 1859 ADDW (R1)+,R2 ;ACCUMULATE THE SUM
0719 1860 SOBGTR R0,10$ ;ONCE FOR EACH WORD
0719 1861 CMPW R2,(R1) ;CHECKSUM OK?
0719 1862 BNEQ 20$ ;BRANCH IF NOT
0719 1863 RSB
0719 1864 20$:
50 0808 8F 3C 0719 1865 MOVZWL #SS$_BADCHKSUM,RO ;ERROR STATUS IN RO
0719 1866 RET
0719 1867
0719 1868
0719 1869 .END

```



FILEREADUV1  
Symbol table

J 10  
- MICRO-VAX I FILES-11 LEVEL 2 FILE READ 10-AUG-1984 18:05:11 VAX/VMS Macro V04-00  
9-JUL-1984 11:44:50 FILEREAD.MAR;1

Page 40  
(18)

```

ARGCNT      = 00000000
BADDR       = 00000401 R      02
BADDR1      = 000004F6 R R    02
BADDR2      = 00000371 R R    02
BADFILNAM   = 00000407 R R    02
BADRET      = 00000406 R R    02
BADRET1     = 00000374 R      02
BOOSGL_RPBBASE ***** X    02
BOOT_UV1_SWITCH = 00000001
BUFADR      = 0000000C
CACHE_ADR   = 00000010
CACHE_SIZE  = 0000000C
CHAN        = 00000004
CHANADR     = 00000004
DIRSB_NAMECOUNT = 00000005
DIRSC_VERSION = 00000008
DIRST_NAME  = 00000006
DIRSW_FID   = 00000002
DIRSW_SIZE  = 00000000
DIRSW_VERSION = 00000000
DIR...      = FFFFFFFF
DIRBUF      = 00000010
DIRNAM      = FFFFFFFEA
DIR_BFCNT   = FFFFFFFFC
DIR_BUF     = FFFFFFFF8
DIR_CACHE_CNT = 00000014
ENTRY       = FFFFFFFD0
ENTRY_ADR   = FFFFFFFF4
EXIT_FILID_FND = 000004C2 R      02
FATSB_RATTRIB = 00000001
FATSB_RTYPE = 00000000
FATSC_VARIABLE = 00000002
FATSL_EFBLK = 00000008
FATSM_NOSPAN = 00000008
FATSW_FFBYTE = 0000000C
FH2SB_MAP_INUSE = 0000003A
FH2SB_MPOFFSET = 00000001
FH2SB_STRUCLEV = 00000007
FH2SC_LEVEL2 = 00000200
FH2SL_FILECHAR = 00000034
FH2SV_BIGFILNUM = 0000000A
FH2SV_CONTIG = 00000007
FH2SV_DIRECTORY = 0000000D
FH2SV_LEVEL2 = 00000009
FH2SW_CHECKSUM = 000001FE
FH2SW_EXT_FID = 0000000E
FH2SW_FID_NUM = 00000008
FH2SW_FID_RVN = 0000000C
FH2SW_RECATTR = 00000014
FH2SW_STRUCLEV = 00000006
FH2SW_VBNOFFSET = 000001FE
FID         = FFFFFFFFA
FIDSB_NMX   = 00000005
FIDSC_MFD   = 00000004
FILSA_DIR_FID = 00000000
FILSA_DIR_OFID = 0000001E
FILSA_IXFRDR = 00000018

```

```

FILSB_DIR_LVL = 00000012
FILSCACHE_INIT = 00000112 RG    02
FILSCACHE_TRUNC = 00000188 RG    02
FILSCHECKSUM   = 00000719 R      02
FILSCHECKSUM1  = 0000071E R      02
FILSCHKFILHDR  = 000006E9 R      02
FILSC_CACHE_ID = 00000001
FILSC_DIR_SIZE = 00000024 G
FILSC_SIZE     = 00000218 G
FILSFINDFILID  = 000002B1 RG    02
FILSGQ_CACHE   = *****W GX    00
FILSGT_DDDEV   = *****W GX    00
FILSGT_DDSTRING = ***** X      02
FILSGT_TOPSYS  = *****W GX    00
FILSL_DIRMAX   = 0000000C
FILSL_DIRNXT   = 00000008
FILSL_DIROFF   = 00000004
FILSL_DIR_BFOFF = 00000018
FILSL_DIR_LBN  = 00000014
FILSL_LBNMAX   = 00000014
FILSL_LBNNXT   = 00000010
FILSL_LBNOFF   = 0000000C
FILSMOUNT      = 0000022C RG    02
FILSOPENFILE   = 0000000C RG    02
FILSQ_DIR_HDR  = 00000010
FILSRDCHKFILHDR = 00000527 RG    02
FILSRDWRTLBN   = ***** X      02
FILSREADVBN    = 000005F2 RG    02
FILSSTATBLK    = 00000685 RG    02
FILST_DIR_NAM  = 00000006
FILSWRITEVBN   = 000005EB RG    02
FILSW_CACHE_ID = 00000000
FILSW_DIR_BFCNT = 0000001C
FILSW_DIR_BKCNT = 00000010
FILDSC        = 00000008
FILHDR         = 00000004
FILID          = 00000018
FILNAM         = 00000008
FIL_GQ_CACHE   = 00000000 R      02
FIL_GT_DDDEV   = 00000004 R R    02
FIL_GT_TOPSYS  = 00000008 R R    02
FIND_LEVEL2    = 0000040D R R    02
FIND_LEVEL2_1  = 00000419 R      02
FM2SS_COUNT2   = 0000000E
FM2SS_FORMAT   = 00000002
FM2SS_HIGHLBN  = 00000006
FM2SV_COUNT2   = 00000000
FM2SV_FORMAT   = 0000000E
FM2SV_HIGHLBN  = 00000008
FORMAT1        = 00000665 R      02
FORMAT2        = 00000678 R R    02
FORMAT3        = 0000064E R R    02
FORMDIRSTRING  = 000001C0 R R    02
GETRTRVPTR     = 0000063F R      02
HDCNT          = FFFFFFFFC
HM2SB_STRUCLEV = 0000000D
HM2SL_IBMAPLBN = 00000018

```

FILEREADUV1  
Symbol table

- MICRO-VAX I FILES-11 LEVEL 2 FILE READ 10-AUG-1984 18:05:11 VAX/VMS Macro V04-00  
9-JUL-1984 11:44:50 FILEREAD.MAR;1

Page 41  
(18)

HM2\$\$\_MAXFILES = 0000001C  
HM2\$\$\_CHECKSUM1 = 0000003A  
HM2\$\$\_CLUSTER = 0000000E  
HM2\$\$\_IBMAPSIZE = 00000020  
INCRS8 00000675 R 02  
INIRTRVPTRSCAN 0000062E R 02  
IOS\_READBLK = 00000021  
IOS\_WRITEBLK = 00000020  
IOFUNCTION = FFFFFFFFC  
IXFHDR = 0000000C  
LBN\_CACHE\_CNT = 00000018  
LIB\$CVT\_DTB \*\*\*\*\* X 02  
NAMBLK FFFFFFFE0  
NAMDSC FFFFFFFD4  
PLACEMENT 0000065C R 02  
PQ = 00000001 R G  
RDWRTVBN 000005F7 R 02  
READ\_DIR\_HEADER 00000375 R 02  
READ\_DIR\_LBN 000004F9 R 02  
RTRVPTRBUF = 00000010  
RTRVPTRLEN = 0000000C  
SAVABS... = FFFFFFFF0  
SCRATCHSIZE FFFFFFFD4  
SCRATCH\_SIZE FFFFFFFD0  
SS\$\_BADCHKSUM = 00000808  
SS\$\_BADFILEHDR = 00000810  
SS\$\_BADFILENAME = 00000818  
SS\$\_BADIRECTORY = 00000828  
SS\$\_ENDOFFILE = 00000870  
SS\$\_FILESTRUCT = 000008C0  
SS\$\_NORMAL = 00000001  
SS\$\_NOSUCHFILE = 00000910  
STATBLK = 00000008  
STORE3DIGITS 000001A7 R 02  
TMPRTRVDSC FFFFFFFF0  
TMPRTRVLEN FFFFFFFF8  
VBN = 00000008

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	FFFFFFFC ( 0.)	01 ( 1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
YFILEREAD	00000732 ( 1842.)	02 ( 2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	14	00:00:00.09	00:00:00.25
Command processing	70	00:00:00.62	00:00:01.13
Pass 1	378	00:00:14.73	00:00:17.13



Symbol table sort	0	00:00:01.93	00:00:02.03
Pass 2	368	00:00:04.82	00:00:06.31
Symbol table output	19	00:00:00.16	00:00:00.16
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	851	00:00:22.37	00:00:27.03

The working set limit was 1350 pages.  
98698 bytes (193 pages) of virtual memory were used to buffer the intermediate code.  
There were 70 pages of symbol table space allocated to hold 1183 non-local and 93 local symbols.  
1871 source lines were read in Pass 1, producing 17 object records in Pass 2.  
23 pages of virtual memory were used to define 21 macros.

+-----+  
! Macro library statistics !  
+-----+

Macro library name	Macros defined
DISK\$STARWORK03:[GAMACHE.UV1ROM.VMS]LIBUV1.ML	6
DISK\$STARWORK03:[GAMACHE.UV1ROM.OBJ]VMB.MLB;3	0
SYS\$SYSROOT:[SYSLIB]STARLET.MLB;2	10
TOTALS (all libraries)	16

1234 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

MAC/LIS=LIS\$:FILERDUV1/OBJ=OBJ\$:FILERDUV1 VMSS:BOOUV1SWT+VMSS:FILEREAD+OBJ\$:VMB/LIB+VMSS:LIBUV1/LIB



0430 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

